



March 1988

Vol. 1 N° 6

Price £1.50

Archive

The Subscription Magazine for Archimedes Users



Fast Screenload / Screensave Module

Graphic Writer Review

Teletext Adaptor Review

More about WIMPs, Pointers and BASIC V

ADFS Disc Editor

The Game of Life

Using the SystemDevs Module

DIY Hard Disks?



Archive prices going up?!

Yes, I'm sorry, but after all that blurb last month about being the best value for money, we've decided that, at £10 for twelve issues, the magazine does not even pay for itself – we are surviving on the software and things that you are buying through the magazine.

So, from the end of March, the annual subscription for Archive will be £12.50 – still cheaper than RISC User (now £14.50) – and still, we feel, better value for money. I think that it is a reasonable price for the amount of material you get – I only hope you agree!

The new air-mail subscriptions are: Europe £18, Middle East £22, America & Africa £25, Elsewhere £27.

Those of you who took out 6 month subscriptions, if you renew **before 31st March**, you can do so at the old rate of £10. And if you have friends who want to subscribe, advise them to do so before 31st March!

Wot, no religious bit?!

Several people noted that we had missed out the “Thought for the Day” bit and wanted to know if it was done deliberately, in response to the negative comments we had received. No, it wasn't that. It was partly pressure of space – we had to use all available space to cram everything in – and it was partly pressure of time.

I was finding it difficult to get all the material ready to meet the printing deadline. I was working almost every evening as well as through the day. Then friends started ‘getting at me’ and saying I was working too hard and neglecting the family. Cheek!... but they were right! So I decided to go back to what I felt God had been telling me some months ago – only to work in the evenings in extreme circumstances and to believe that He would enable me to get all the work done in time.

It worked! The magazine is just about to go to the printers half a day ahead of schedule. People can say what they like, but I believe that God does work miracles – and I don't just mean in helping me with my work...

Archive

Volume 1 • Nº 6 • March 1988

Contents

Hardware + Software Available2	DIY Hard Disks?26
Archive Binders are here!4	5-Byte Time Format.....28
Matters Arising4	WIMP Environment – Part 430
Help!!!.....5	The Game of Life.....32
Hints & Tips6	More about Pointers.....36
Information about Monitors 10	More about BASIC V37
Bug or “Feature”? 10	Floating Point Emulator39
Reference Manual Errors 11	Using the SystemDevs Module...41
GraphicWriter Review 12	ADFS Disc Editor43
SCML Teletext Adaptor 14	Order Form51
NewsMaster Review 16	*HELP Archive.....52
Fast Screenload/Screensave 19	Fact-File53

Hardware & Software Available

- **CCD Computer Services** are offering three new pieces of Archimedes software: A **graphics library for Fortran** (£49.50), a **text editor** for compiled languages with full screen editing, an editing command language and access to the operating system, handling 5,000 lines of text (£19.50), and a **printer buffer** module (£12.50).

- **Font Designer** – an editor for BBC fonts on the Archimedes. (N.B. 8 x 8 characters, not fancy fonts) For £8.95, APL Software send you an editor, manual and 15 sample fonts including scientific symbols. We've got a review copy and it looks quite good – you can invert, flip, rotate, shift, angle, copy etc – all mouse-driven.

- **Archimedes Logo is here!** Logotron have released their full Archimedes version of Logo (i.e. it does not run under the 6502 emulator). It is claimed to be over 10 times faster than BBC Logo, will use all Archimedes graphics modes plus a 1024 x 960 pixel graphics resolution mode. It gives program control over error handling so that the level of help offered by error messages can be adjusted to suit students' needs, it supports the FPU and has a multiple screen turtle control mode. For £60 you get reference manual, tutorial manual and example files. (Note that extension modules written in 6502 code will NOT be compatible. Logotron say that various Archimedes extension modules will appear during 1988.)

- **4096 colour screen dump** for Integrex 132 colour ink jet printer. Musbury Consultants (!) have produced a screen dump which will cope with the full 4096 colours that Archimedes can generate. A full screen takes about 4 minutes to dump and the results are quite impressive from the samples they sent us – even the aspect ratio is correct – circles are circular! The full price is £30 but they are prepared to offer a discount through Archive magazine, so if you send your orders in to us, you only need send £25.

- **Maths Movies...** Manchester Polytechnic has made some computer 'movies' to help students appreciate the subtleties of things like fourier transforms. The transforms of moving two-dimensional shapes are calculated and re-played as slowly or as quickly as you like – up to 5 frames a second. Each disc, costing £10, consists of two movies, each of 21 frames, of specific moving objects, e.g. growing squares, rotating bars and grid patterns. The programs need a 1M machine and each disc comes with a text file explaining the theory of the transform and special points to note with the particular movies. For a full list of movie discs, write to Dr J Slater, Dept. of Maths & Physics, John Dalton Building, Manchester Polytechnic, Manchester.

- Although not specifically for Archimedes, those of you with an Archimedes and perhaps a BBC and who are rich enough to own TWO printers(!) will like to know about a "**Cross Switch**" by **Data Switches** available from Datastore in Bromley. It has a single switch which swops over the two printers between the two computers. It costs £66.95, but presumably you would also need to buy two Centronics-Centronics cables to link to each of the printers.

- **Graphics Library.** Micro Studio are launching a three disk set of colour graphics pictures for inclusion in programs or for use with Artisan or GraphicWriter. This first set of disks (£21.95) should be available by the time you read this and others will become available through the year. The sample disc they sent had some very good animal pictures and a few others that were very clever but a number of the others were what I would call 'idiosyncratic', amongst which I would include the head of a gentleman with a black beard and horns against a red background! (To be fair, there was also a picture of Jesus, breaking bread!)

• **Database Management System** – Flying Start II (“Records, Files, Lists and Labels in an Instant”) is now available from Mitre Software to work on the Archimedes (using the PC emulator). Price £69.95.

• **“Desktop Games”** is the title of the first offering from Gem Electronics who specialise solely in Archimedes and aim to produce “quality software at sensible prices”. For just £5.95 inclusive, you get Mastermind, Sliding Block Puzzles and Solitaire as add-ons to your desktop. All three have different levels of difficulty and you can use your own pictures (created with Artisan, Paint etc) with SBP.

Even if you don’t really want the games, it might be worth the £5.95 to get the other utility that comes with it which enables you to enter *-commands from within the desktop!

• CJE Micros have a **disc interface** for an external 5.25" drive – £30, quite neat, says one reader, has link settings so that you can swop the internal and external drive identities, i.e. make the 5.25" drive into drive 0 – very useful with the PC emulator for running protected software.

• Ace Computing are about to launch ‘Euclid’, their **3-D graphics animation system**. (End of March, hopefully.) The idea is to provide ARM code routines which a relatively inexperienced programmer could access from within BASIC. Objects can be built up from a library of simple geometric shapes. Re-drawing, from the sample we’ve seen, is extremely fast (written in ARM code) so that you could get smooth animation of simple shapes, zooming in and out and rotating around, even without using shadow screens. It will come with a basic library of shapes and users are encouraged to share the shapes they generate with other Euclid users. Details from Tony Cheal at ACE Computing.

• **Computer Concepts ROM Podule** We haven’t actually got any ROMs to put in it yet, but what we are getting excited about is the RAM filing system which allows you to do all sorts of useful things.

The RAM filing system is like the ADFS in many ways though obviously you cannot use *DRIVE, *FORMAT, *TITLE, *VERIFY, etc) – you can, however, have directories, you can *COPY things and you can put your favourite utility programs in it. It’s great!

To copy between the ADFS and RFS, for example, you might say:

```
*COPY ADFS:$.file RFS:$.file
```

For me, the best facility is being able to have all my own special little programs in the RFS so that they are always available – as if they were in the LIBRARY directory on the current disc except that they are always available even when you change discs. To set it up you have to set the ‘filepath alias’ so that the Archimedes looks in the RFS as well as the ADFS library for files. You might think that you would therefore have to set this up every time you switch on the machine – perhaps by using a disc with a special boot file. Not a bit of it!


You can set the Archimedes to boot up in the RFS by altering the configuration. The RFS boot program can then set up the filepath alias (and other aliases that you use regularly or function key definitions) and then return to the ADFS and continue as though nothing had happened. That way you can have your RFS library and your own aliases permanently installed as if they were part of the operating system. So even when you do a <ctrl-break>, say after *configuring something, it restores all your aliases and function key definitions!

The ROM podule with battery-backup is £63 to Archive members and the 32k ram chips are £9.50 each (128k chips are not currently available.) The podules are becoming available, but demand is far out-stripping supply, so all dealers are being strickly rationed. If you want one, send in your order and we’ll hold onto your cheque until we can supply them. **A**

Archive Binders at last!


Many thanks to those readers who responded to our plea for a source of good quality A5 binders. You can now order your Archive binders for just £5.50 including UK postage and packing. You can now keep all your Archive magazines together in one place for quick reference – as I can testify, having had my Archives bound for a few weeks now, it makes life a lot easier to be able to flick through the different issues of Archive when looking for some particular article or piece of information. (We hope to have news next month of the availability of a 6-month index which will also be a great help.)

Fitting the magazines into the binder is simplicity itself – even I can do it! And they don't fall out even though the binder is not actually full.

Archive Binders £5.50 available only from Norwich Computer Services. 

Problems with linefeeds

When will the software producers learn?! Those of you who have tried to use various different pieces of professional software will no doubt have discovered that some assume auto-linefeed is ON while others assume that it is OFF! To save us all from going mad trying to dig inside our printers flicking dip switches to and fro, why doesn't everybody send linefeeds so that those who don't want them can configure their Archimedes' to IGNORE 10's and those who do want them can IGNORE 0's – simple, isn't it?

Please, software houses, act now so that we can 'nip this in the bud'! 

Matters Arising...

• **Fact or Fantasy...?** We are warned by someone who has inside knowledge of the pcb layout that the boards for the 300 and 400 series are sufficiently different to make it impossible to do an upwards conversion. For example, the connections to the 96 way connector are just not available.

As regards replacement of the ram chips, someone else tells us that all the lines on the existing chips seem to be used so he doesn't see how it could be possible to have pin-compatible chips with four times the memory. Anyone any ideas on that?

Another source warns us that the pcb is a multi-layer board, so the contact between the layers is done via plated-through holes. This plating is apparently not too secure so that attempting to desolder components without using the proper vacuum-desoldering equipment could prove fatal. You cannot reconnect to the intermediate layers, so if you strip the plating off just one hole, the board may well be a complete write-off! Don't say we didn't warn you!

• **BASIC / OS Interface.** A minor point of correction to Gerald Fitton's article in Archive no.5. (*minor, but interesting, none-the-less...Ed.*) The Acorn ARM CPU doesn't have a 'microcode' program: assembler statements are coupled through to the transistors in the chip pretty directly, rather than being run by some hidden internal program (unlike an 80286 or a 6502) – this is one of the key reasons why RISC machines are so fast and efficient. For those who are interested in the internal hardware, there's a good article in the June '87 issue of 'Electronics and Power' (published by IEE), which also talks about the MEMC, VIDC and IOC chips.

Also, on the subject of the BASIC/OS interface, users of the BBC model B / Master with 6502

2nd processor who want to run BASIC V, might like to be reminded of Tubelink's 'Advanced BASIC'. It recognises most BASIC V keywords (and tokenises them correctly) and also has a pretty complete implementation of the 'SYS' statement for OS_Byte, OS_Word, etc calls that can be emulated on a 6502 machine. **A**

Help!!!!.....

• ID Campbell has an **Acorn Teletext Adaptor** and asks whether it is possible to hang it on the I/O podule. Acorn say not, but someone must be able to get it going. If so, let us know. Thanks.

• In reply to the help plea about **VDU23,26**, Keith Milner says that the VDU driver needs the ASCII codes for the characters of the font name, e.g. ASC"C", ASC"o", ASC"r", ASC"p", ASC"u", ASC"s" at the end of the VDU call, but that it is easier to do it with VDU23,26,1,0,0,160,160,0,0:PRINT"Corpus"

He also says that you should not be using it anyway! The VDU call is obsolete – only there for upwards compatibility. You should apparently be using SYS"Find_Font". (Someone else pointed out that the font names should have a special prefix, which is why it could not find them.) I really think we need someone to explain this all to us properly. Any offers?

• Does anyone know if the **Morley RAM disc** works on the Archimedes 1MHz bus? says R.A.Brown, Tadworth, Surrey.

• Could someone write us an article about **how to convert old BBC programs** to get the best out of the Archimedes, please?

• Wanted: an **ARM language sort routine** to selectively sort an array of fixed length string. The input parameters, transferred from BASIC would be (A%,B%,C%,D(0)) where A% = no. of elements of the array to be sorted, B% and C% = first and last characters of the string to be compared, D(1) = the first array element to be included in the sort. K J Gardner of 7 Brooklands

Avenue, Preston, PR4 2BW, who asks for this, says he has a BBC version taken from Stephenson's "Advanced Machine Code Techniques".

• **Problems with the PC Emulator?** If you have more than two floppies attached, the software apparently won't recognise the extra one(s). The enquirer has set up DRIVER.SYS in the CONFIG.SYS file and it loads the external driver for drive D but uses drive A: or B: prompting him to change discs.

Also there is a rumour, admittedly spread by an Amiga owner trying to knock Archimedes, that CHKDSK can corrupts discs. Any truth in it?

• One problem with both CJE Micros and Dudley Micro Services **5.25" disc interfaces** is that when you access the internal drive, the external one starts up too. It doesn't do anything to the disc, as far as we can tell, but it's disconcerting when both drive lights come on. This can be avoided, says one reader, by going inside the 5.25" drive and removing the links marked MM and IU and putting one of the links back on the MS pins. OK, fine, but my drive, a Mitsubishi mechanism, I think, has got links marked MX, HH, HL, HM, HC, HS, RR, UD, IS and IL!!! Any suggestions anyone?

• **Avoiding DATA in library procedures.** All the extra structuring commands provided in BASIC V enable us to avoid the use of line numbers – thus enabling the use of libraries of procedures and functions. However, READ and DATA cannot be used in a library procedure because you cannot use RESTORE since it refers to a line number. Any ideas how we can get round this problem?

• We need a **RAM disk** on the Archimedes. There isn't one in OS 1.2 and we can't wait until 2.0 appears (in a year or so's time?!) There must be someone out there clever enough to write one. Is there? (If you want one now, you could buy a backplane & CC's ROM podule plus some ram to go in it, but it's not cheap.) **A**

Hints and Tips...

- (1.2 OS) If you want to look at the **desktop manager programs** and perhaps modify them and run them from disk, the following information may be of use:

The DESKTOP has its own filing system activated by *DESKFS. If you do a *CAT, you will get:

```
DeskTopMgr
DeskTopMgr2
&.!palette
Icons.Desktop
Icons.Calculator
Icons22.Desktop
Icons22.Calculator
```

You can then LOAD the DeskTopMgr2 program and LIST it. (DeskTopMgr is just the 5-line startup program that gets left in BASIC when you exit the desktop.)

If you change back to *ADFS, you can then save the program. To copy the palette and/or the Icons, you have to use commands such as:

```
*COPY DESKFS:I*.Desktop $.I*.Desktop
```

(assuming you are in ADFS and have created a directory called Icons)

To run the program from disk, you need the icon and palette files to be copied across then if you fancy tackling 76k of “crunched” BASIC program, you can try editing it!

- If you are using the **decrement or increment** (\pm or \mp) in a BASIC program and you get a “Mistake” error on that line, the chances are that you have forgotten to initialise the variable. Try typing `X=X+1:PRINT X` and it will give the value 1 but if you type `Y+=1` it will say “Mistake”. What it really should say is “Unknown or missing variable”, but still, good programmers won’t get the problem because we always remember to initialise all our variables, don’t we? (We do?)

- **Testing multi-sync output** without a multi-sync monitor! If you are writing software and hoping to sell enough of it to be able to afford a multi-sync monitor(!) you have the problem of testing your software in modes 18 to 20. If you use *Configure MonitorType 1, modes 0 to 17 are still displayed as normal then when you select modes 18 to 20 and generate the screen output, you can use a screendump (such as Gerald Fitton’s minidump or *HardCopyFX) to look at what you’ve got. If you’re worried about damaging the monitor, switch it off before changing mode.

- **Using large printer buffers** etc. If you’re using a buffer that’s larger than 64 kbytes, it’s no use trying to use ADVAL with a negative number (e.g. `PRINT ADVAL(-4)` for the printer buffer) – it only returns the least two significant bytes. Instead, use `SYS “OS_Byte”, &80, &FC TO , X%, Y% : bytes_free%= X%+256*Y%` where &FC is the -4 value. (*Actually, instead of working out that -4 is the equivalent of &FC you can, it seems, be lazy and use -4 in the SYS command. Ed.*)

- **Easy loading of modules:** To enable you to load the emulator and fast BASIC easily, you can create some library programs on your Welcome disc (or better still, a copy of your Welcome disc!). This is what you do for fast BASIC:

```
*BUILD LIBRARY.FAST
```

```
*FX225,1|M
```

```
*KEY1 QUIT|M*MODULES.RAMBASIC-
      |M*KEY1|MCLS|MHELP|M
```

```
*FX138,0,129
```

```
<escape>
```

and then change it into a BASIC program with

```
*SETTYPE LIB*.FAST &FFB<return>.
```

Note that there are double pad characters before each ‘M’ in order that when the program line is

run, each double pad character is interpreted as a single pad character for the actual key definition.

The bits after calling the module are optional. The first is to clear key1 again in case you press it accidentally. The second is to clear the screen and the third is to give a help message to show that you are actually in RAM_BASIC. (See below).

Once this is set up, to get into fast BASIC all you do is type `*FAST<return>`.

For the 6502 emulator, use the name `LIB*.6502` and don't put the `HELP` command in because that does not work on BASIC IV.

• **Tidying up after ArcWriter!** You can do a similar thing to the above hint in order to tidy up after using ArcWriter.

```
*BUILD LIBRARY.RESET
*FX225,1|M
*KEY1 QUIT|M*CONFIGURE FONTSIZE
      2|M*RMREINIT FONTMANAGER|M
      *RMTIDY|M*BASIC|M
      *KEY1|MHELP|M
*FX138,0,129
<escape>
```

Typing `*RESET` will do the tidy-up for you.

• **Identifying RAMBASIC.** To get RAM_BASIC to tell you that it actually IS the RAM version and not the ROM version, you can change a few bytes so that it prints out 'RAM BBCBASIC' instead of 'ARMBBCBASIC' on the startup and HELP messages. To do this, proceed as follows:

```
*LOAD MO*.RAM* 20000
?&202B0=ASC"R"
?&202B1=ASC"A"
?&23F78=ASC"R"
?&23F79=ASC"A"
*SAVE MO*.RAM_BASIC 20000 +E434
*SETTYPE MO*.RA* &FFA
```

• **Reversing the CAPS lock.** If you hold down the shift key when you put the CAPS lock light on, you will find that the shift key now has the opposite effect, i.e. letters typed without pressing shift come up as uppercase letters, but when you press the shift key you get lower case letters. This can be useful if you are programming and mostly work in upper case but with occasional use of lower case.

• **Displaying screens in different modes.** If you have a screen that has been SCREENSAVE'd, you can display it in another mode. (e.g. Artisan screens in 256 colour modes!) Assuming you have configured spritesize to at least 11, proceed as follows:

```
MODE 13
*SLOAD filename \ NOT
                                *SCREENLOAD
*SCHOOSE screendump
PLOT &ED,0,0
```

You will note that the palette has changed and that it is a lower resolution, however, it will still be quite decent and you will have all those lovely colours to play around with!

If you choose mode 15, you will get the same picture displayed in 256 colours but squeezed into only half the screen width.

If you want to change the colours you will have to load it into the sprite editor (SEdit) and change the colours manually.

• **Using the View Series.** You will find that after coming from the desktop, TAB does not work. The solution is to type `*FX219,9`.

ViewProfessional works fine if you use the second processor version – the VP file on the 5.25" disk – however it should be `*LOAD`'ed at 4000 (not 8000) and called with `*GO 4000`. (See page 5, Archive 1.1)

• **Diary/Notepad problems.** Beware: If you are printing something from notepad or diary, check that an active printer is on-line before starting to print otherwise the system may hang up and the only way out is to press <ctrl-break> – frustrating if you haven't saved the text first!

• **Slower listings.** I know this is obvious to ex-BBC users, but I've not seen mentioned anywhere the fact that holding <ctrl> and <shift> keys down together will stop the screen scrolling. Archimedes lists programs so quickly that even if you use <ctrl-N> to get a paged listing, you may well get two pages instead of one if you hesitate on pressing the shift key, so using <ctrl> and <shift> allows you to control the listing yourself.

• **File transfer between wordprocessors.** If you have Wordwise Plus or View files and you want to import them into GraphicWriter, use *SETTYPE filename &FFF to ensure that GraphicWriter sees them as text files. (Actually it should already see Wordwise Plus files as text because of the save address that Wordwise Plus uses – check by doing *INFO *.) Then you can simply load the files into GraphicWriter using the file menu. To transfer the files back to View or Wordwise, you need to save them as ASCII files. This is done by opening the file menu then clicking on the word "TEXT" with the ADJUST button, typing in a filename and pressing <return>.

(I tried to do the same sort of thing with ArcWriter but it kept crashing each time I tried to load a file, so I gave up!)

• **Smart ON ERROR routine.** If your ON ERROR routine includes the following, you get the usual error message and then, if you press any key other than <escape> it will drop into the ARM BASIC editor at the offending line. So if you do not want to go into the editor, press <escape> instead.

```
ON ERROR OFF
PRINT REPORT$+" at line ";ERL
dummy%=GET
SYS 5, ("KEY0 EDIT "+STR$(ERL)
      +"|M")
SYS 6,138,0,128
END
```

If you prefer to move into EDIT a few lines above the offending line, use STR\$(ERL-50) or whatever.

• **Dual purpose boot files.** It is quite possible to have a boot file that will work equally well from within the desktop or as a conventional boot file operated with <shift-break>. What you do is *BUILD the file as usual but then *settype <filename> &FFB which gives it a link to BASIC. If you then double-click on it in the desktop, it adds line numbers to it and runs it as a BASIC program. However, it will only work with <shift-break> if the machine is configured to start up in BASIC, not in the desktop (i.e. *CON. Language 4, not 3).

• **With the PC emulator** you can get more space than Mark Sealey reckoned in his review last month by *unplugging various of the modules. On reader reckons to get almost 590,000 bytes free (= 576k).

• **Disabling Modules.** Archive no.5 explains how *Unplug can be used to disable Modules, but this method needs a <ctrl-break>. An alternative is to use *RMKill and *RMTidy from the operating system *-prompt (not from BASIC), e.g.

```
*RMKill FontManager
*RMKill StringLib
*RMKill Percussion
*RMKill ARMBasicEditor
*RMKill WindowManager
*RMTidy
```

This typically releases 96 Kbytes of memory on

an A310 machine and doesn't need a <ctrl-break>. It is particularly useful if you're short of space in BASIC, Pascal or FORTRAN, etc and aren't using fancy sound, the WIMP environment or the BASIC editor. Each module can be reinstated by using *RMReinit <Module name>, or all can be restarted by <ctrl-break>.

- **GW BASIC on the PC Emulator.** There is an undocumented way to get back into the MS-DOS operating system after using GW BASIC. Try pressing function key f-11.

- **PC Emulator.** Version 1.09 (which existing users can get by sending back their old disc plus £15 to Acorn) runs, amongst other things, Ability Plus, dBase 3 Plus, Kermit, Word Perfect, LTS NewsMaster (see separate review). (Can anyone add to this list?)

- **RS423... yet again!** If you have the 1.2 operating system and if you have the version of the serial chip that is made by GTE (it's the 28 pin IC at the back left of the pcb) and if you change over from using the CTS line at the Archimedes end to using the DSR line (i.e. link 1, 4 and 8 and join pin 6 to the RTS line from the other machine) then Acorn have a fix which you load in as a module. Send a blank disc to Customer Services to get a copy. The version they sent me didn't work the first time I tried it with some data transfer software so I pressed <escape> and tried again and it worked perfectly at 19,200 baud. Then I tried again and it wouldn't send anything, pressed <escape> and away it went! Still, when you do get it going it sends and receives at 19,200 baud in both directions without any data corruption. I left it running for a couple of hours and there was no corruption at all.

- **After the desktop...** simpler than the list of FX commands that we gave last month is to call SYS"Wimp_CloseDown" (which is what Acorn should have put in their desktop program

in the first place!!!) but if you don't fancy trying to type that every time, getting upper and lower case exactly correct, you can create a one line BASIC program with 10 SYS"Wimp_CloseDown" and save it on your disk as \$.library. undesk and then just type *undesk (or whatever abbreviation you can get away with) and all will be restored to normal.

- **Using the Seikosha 250X printer** is a bit of a problem. When you plug it into the Archimedes, all power is lost. This seems to be because of the +5.0 volts on lines 18 and 36 of the printer connector – these are earthed at the Archimedes end and so have to be disconnected somehow. One reader, Adrian Moreton, has succeeded but it is too complicated to explain here. If you want the information, send us an s.a.e. and we'll send you a photocopy of his letter and the excellent diagrams he has drawn.

- **ADFS wildcards.** Had you gathered that ADFS will take * or # wildcards **anywhere** in a filename? On the BBC, you could only put them at the end of the filename. So, for example, if you've got PROGRAM1, PROGRAM2, PROGRAM3 etc you only need to say something like LOAD "PR*2" or even LOAD "*2" if none of the other files have numeric endings. That makes life much easier, doesn't it?! (Thanks to Benjamin Finn for pointing that out. The next hint is his, too.)

- **Errors in library procedures.** If an error occurs in a PROC or FN that is installed using LIBRARY or INSTALL, the correct error message is given but the line number quoted is the last number in the main program. Although we cannot get a line number pointer to the error, we can at least find out from which procedure in the main program the library routine was called when the error occurred.

At the beginning of each definition of a PROC or FN you should put something like:

```
DEF PROCdosomething
LOCAL ERROR
ON ERROR LOCAL PRINT REPORT$
    " in PROCdosomething":END
:
ENDPROC
```

• **Help with Econet** is offered by Michael Ryan of XOB, Balkeerie, Eassie, By Forfar, Angus, DD8 1SR. Michael says, "Here are some comments about Archimedes on Econet in response to comments in Archive 1.3, page 8.

"RDFREE is part of the Econet filing system module and hence not a disc-based utility as with the BBC version. Issued with a user identity as parameter, it gives the total disc free space and that user's allocated portion of it. Issued with no parameter, it returns the user's own free space and the total free space.

"Acorn are supplying a number of other utilities including SETFREE, SETSTAT, USERS, PROT and UNPROT, though they have not yet made it clear how they are to be distributed, so don't hold your breath.

"Though it is very unlikely that they will supply the more sophisticated utilities (VIEW, REMOTE, NOTIFY etc) all is not lost. XOB already have a wide range of utilities for the BBC/Master series and have already implemented several of them on the Archimedes. More details from us on 0307-84364." **A**

Monitor Information

• **"Taxan 770+ multisync monitor** works very well, 132 column text is easily readable and it supports the three high-resolution modes."

(We're hoping to get hold of one for review and maybe to see if we can do a deal with Taxan to buy them for members. If you are interested, drop us a line.) **A**

Bug or Feature?

• If you prepare your **BASIC** programs using a text editor and say ***BASIC -LOAD <filename>**, it takes the file as if it had come from the keyboard and numbers it as a BASIC program. However, with large files, it sometimes corrupts the end of the program. (BASIC 1.02) For example, create a file of 1400 or more lines of text all saying "HELLO", save it as HELLO and then do ***BASIC -LOAD HELLO** and then **EDIT** and have a look at the end of the program. You will find that the program starts to go wrong at line 13660 and all lines beyond 13790 are lost completely. Any comments Acorn?!

(The solution is to put **AUTO** as the first line of the text and then ***EXEC <filename>**.)

• **ArcWriter**. Those of you who have received your copy of Acorn's amazing free word-processor will realise that there are, to be generous, 'one or two' problems with it. I was going to list them to make the point that, whether the software is 'free' or not, this is not the sort of standard we expect from a company like Acorn. However, there are so many bugs it seems hardly worthwhile.

One problem that it worth mentioning, simply because there is a solution to it, is the buzz! Several of you have noted that when ArcWriter is running there is a continuous whine, in some cases accompanied by sound-on-vision effects. This can be eliminated or at least considerably reduced by soldering a 100µF capacitor onto IC 86. This was described in Archive 1.3, page 7.

I had problems originally trying to get it NOT to double space on my Panasonic KP1081 but after using the **AW_PCedit** program to change the end of line code to 13 and the changing the dip switches to put the auto linefeed OFF it was OK. It's a bit of a pain though because I use the same

printer on the BBC micro with a data-switch and the BBC wants auto linefeed ON. The other problem is that if I forget to switch the printer over to the Archimedes and try to print the text, ArcWriter hangs up and I lose all the text!

I'm sure you've all got some tales of woe to tell about ArcWriter, but please don't send them to me. Until we get a version of ArcWriter that at least NEARLY works, I'm not prepared to clog up the pages of Archive trying to tell folk how to get it going.

The answer is to buy Graphic Writer from Clares which is, in my view, under-priced at £29.95 for the facilities it offers. (See review on page 12)

- **OS_BreakPt** does not seem to work properly. The PRM (pages 341/2) says that the default handler prints the register contents and enters the Arthur Supervisor, but apparently although it used to do so on 0.2 OS, it doesn't on 1.2 OS. Acorn say this is a feature, not a bug!

- A couple of readers have spotted the **problems with TRACE**. If you try the following program you will see that it misses out various line numbers:

```
10 REPEAT
20 PROCTest
30 UNTIL 0
:
100 DEFPROCTest
110 PRINT "Hello ";
120 ENDPROC
```

All the trace manages to come up with is [10] Hello [30] [120] [10] etc, i.e. it does not make any reference to the procedure call (20), the procedure definition (100) and the first line after the definition (110). Any comments Acorn??!

- ***FX25** in previous operating systems resets ALL the character definitions to the system defaults but in OS 1.2, characters 128 to 159 are left as they are. So the statements in the User Guide and the Programmers' Reference Manual

that **"*FX25,0 restores characters 32 – 255"** is not correct. To restore characters 128 to 159 you should be able to use ***FX25,4** but that doesn't work either!!!! Try:

```
FOR N%=128 TO 159
VDU23,N%|
NEXT N%
```

(Note the pad character after the N% to pad out the VDU command with zeros.) **A**

Programmers' Reference Manual Additional Changes

Here are some changes in addition to those already given in the Addendum that comes with the Reference Manual.

- p. 37, para 3 should read "When a pixel is plotted, the following occurs, in terms of the actual colour stored in the screen (and returned by the OS_ReadPoint call:- the bottom six bits of the colour number (set by VDU17 and 18) are shifted; bits 0 and 5 two places to the left, bit 4, bits 1,2 and 3 three places to the left and bit 4, one place to the right. The appropriate tint value is shifted six places to the right, into bits 0 and 1, and then the two parts are combined."
- p. 209 ***CONFIGURE File** should be ***CONFIGURE FileSystem** and it will accept names as well as numbers
- p. 216 The note at the top of the page is untrue!
- p. 583 the syntax of the Co-Processor Status Transfer is down as **op<cond>prec<round> Fm, Fn** but it should be (according to one reader) **op<cond> Fn, (Fm:#value)**. (I can't check that – it's all greek to me!)
- Corrections on corrections! In the Addendum for use with Issue 1 PRM, on page 6, for "Page 22: VDU 22,n" should read "Page 82: VDU 22,n". **A**

GraphicWriter Review

Jonathan Marks

My intention here is to look at the facilities provided by Clares Micros' new wordprocessor package for the Archimedes. Although this is not intended to be a comparative review, having also just received ArcWriter, it is difficult not to make comparisons. However, having not gone into ArcWriter very deeply, I cannot call this a full comparative review and in any case, is it fair to compare a 'free' wordprocessor with one for which you have paid out £29.95?

The stated aim of GraphicWriter is to provide an 80-column wordprocessor with characters printable in plain, bold, italic, underlined or any combination thereof. It allows the inclusion of graphics, mixed in with the text, the pictures either being created in GraphicWriter's own graphics editor or being imported from other sources such as Artisan or graphics libraries. *(Such as the one mentioned in the Software Available section. Ed.)*

It allows you to have several documents in memory at once and you can transfer blocks of text between the different documents. Although each individual document can only be up to 32k characters long (roughly 5,000 words or 10 pages of typed A4) you can make up a larger document (subject to total memory available) out of several of these mini-documents and they can be printed out in succession with page numbers etc all taken care of.

Getting started

If you put in the GraphicWriter disc and press <shift-break> you find yourself in an Inter-Word-like environment with some sample text which is loaded automatically. You very quickly find that the text cursor can be moved around with the cursor keys and that the scrolling takes place at a terrific speed (in marked contrast to ArcWriter!). Also, if you try typing, you find that there is no way you can beat GraphicWriter for speed. (Another contrast with ArcWriter.)

A pointer is visible on the screen, so you try it and find that you get a little blue character block each time you press <select>! Pressing <adjust> moves the text cursor to wherever you point, though it is a bit disconcerting because the document scrolls so that the insertion point is (as you later discover it always is) just above the middle of the screen. The <menu> button brings up some incomprehensible icons, so you decide it is time to look at the manual!

The manual

To be honest, this is where the trouble starts. There is a saying that "a good picture is worth a thousand words". Sadly, the author of the GraphicWriter manual does not apparently subscribe to this view – there is not a single diagram or screendump in the whole manual – just 60 pages of solid text, typewritten and (?)photocopied, though at least it was done on a daisy-wheel printer, not a dot-matrix.

Having said that, it won't take you long to get used to the way GraphicWriter works – it's common sense mostly – the manual just provides information for when things are not obvious from the on-screen prompts.

For the rest of this review, let me just go through some of the facilities, making comments about what is provided.

Search & Replace

This is very versatile – global or selective – case sensitive or not – find the next match (strangely called 'continue' rather than 'find') – change this match – change all matches – change from the cursor to the end of the document – change this match and exit to edit mode.

How is it for speed? It goes like greased lightning! In a 28k document it changed every letter 'a' into '@@' so fast that I couldn't time it – under a second, anyway.

One negative comment though, and this applies to a number of facilities within GraphicWriter, is that if you are searching for a word and it cannot find it, it produces a rather intimidating 'Error Window' telling you that the 'System Reports Error: &10007'. It makes you think the software has crashed or something equally horrendous. It also says 'Word(s) not found' – "Oh, is that all?!", you sigh. Surely they could have provided a more friendly response to what was, after all, a quite reasonable request.

The search and replace is available either from the menu via the mouse or by pressing a function key (though there is no function key strip provided!) and you can also use a function key to find the next occurrence of the search string – the manual refers to this as 'go to last search string' which I would have thought meant going back to the previous one! This makes searching through the document very quick and easy.

Another grumble here though is that if you try to change the search string, you find that when the window comes up they have put the cursor at the **beginning** of the search string. You then have to move it to the end and use the delete key to delete it before typing in the new search string.

Preview

Two types of preview are provided – minipreview and full-size preview. The former gives you an on-screen window showing a view of the whole page at a single view. There is a button to allow you to flick through the pages and icons to allow you to print or go to a full preview of the selected page.

The full preview gives you a WYSIWIG full-size view of the page and moving either the cursor or mouse, slides you (and I use the word 'slide' advisedly!) up and down the page so that you can look at the detail that the minipreview does not show.

Sprites and Graphics

If you want to put pictures of any sort onto your documents you can either import pre-prepared

sprites – say from Artisan or other art packages – or draw diagrams within GraphicWriter's own graphics mode. You have circles, ellipses, sectors, chord segments, rectangles, squares, parallelograms, fill, triangles, individual lines and linked lines. The graphics can be scaled to fit into the space available for them on the page and this can be done dynamically in the page composition mode.

Summary

GraphicWriter is well named because it is more than a wordprocessor since it allows the inclusion of pictures. It is not however anything approaching a desktop publishing system such as the one Computer Concepts are working on. It is simple enough to use as a straight forward wordprocessor and even with the addition of pictures it is not too difficult to understand. My only reservation is the quality of the manual, though that is not too much of a problem as the software is so easy to drive anyway.

Overall, I think Clares have under-priced GraphicWriter. They set the price well before the software was ready for sale and I suspect that it suffered from 'creeping featurism' – you know, as you develop a program someone says, "That's good, but it would be nice if it could also..." and so the program grows a bit more. By the time GraphicWriter was ready for sale, I guess they would be a bit cross that they hadn't decided on £39.95 instead of £29.95 because I think it would still sell at that price. They may have been worried about ArcWriter – because it was to be provided free of charge. They need not have worried, as those of you who have tried ArcWriter will probably realise!

One point on which I must commend Clares is their policy of not protecting their software. Yes, it opens it up to the dishonest people who will inevitably copy it illegally, but it does make life easier for those of us who are prepared to pay for good software – it enables us to keep a copy of the wordprocessor on each disc that contains GraphicWriter text **A**

The SCML Teletext Adaptor

Matthew Treagus

What is Teletext?

Teletext is a free Viewdata system available to people with a special adaptor on their television or computer. It is a form of one-way communication where the pages are sent in series and not at the specific request of the user. This limits the number of pages that it is practical to provide because you need to balance the number of pages available with the time taken to access a given page. Presently it can take about 30 seconds for a teletext page cycle to come round again to the same page. Each page may be split up into sub-pages which are normally updated every other cycle.

Teletext for the Archimedes

SCML, as a sister company to Solidisk, have again emerged ahead of the competition, as they so often seem to do, with the appearance of their Universal Teletext Adaptor to run on the Archimedes. The theory behind their system is that it has a standard unit with Aerial Input and both a Data Output to the computer. The appropriate software is then written for the new computer and a new lead made – this must surely be the cheapest and most profitable way to market hardware. The Units also have an Auxilliary Output for direct connection to a composite monitor (such as the Philips CM8833) which would allow it to be used as a television but with the monitors supplied with the Archimedes this is not possible because they do not have composite video inputs.

The Archimedes Software installs itself as a relocatable module and the unit plugs into the printer port. This may prove a slight disadvantage to those who wish to obtain hard copies of the Teletext pages. The alternative would be to use a serial printer, or else the

screens could be saved to disk and printed later. (*Or perhaps you could use a Data Switch. Ed.*)

The pages are recalled from disk by a BASIC program which it would be extremely easy to customise. The adaptor needs to be plugged into the mains and a good TV signal is essential.

Tuning the Unit

The unit is controlled totally electronically from the software, including the channel switching and tuning. Tuning is done by selecting your local transmitter from a list of 64 and the fine tuning to within 0.05 Mhz of the signal. This can then be saved as a “customised” version of the TTX module.

Commands

The commands, as follows, are accessed through single keypresses and makes for an easy-to-use, fast system.

- C – Selects Channel (1 to 4)
- F – Fine tuning to 0.05 MHz
- H – Hold page stops a page being updated
- P – Telesoftware Program Catalogue
- R – Reveal
- S – Save page
- T – Tune to new transmitter
- @ – Select “fast-text”

Telesoftware

I have regularly downloaded the Telesoftware every week and have obtained many very useful pieces of software from the service including assemblers, games, KERMIT and Micro User listings. There is a facility for automatically downloading all the software in one go as the “Auld” teletext utility did on the BBC machines. The down loader software however does not retry when errors occur but aborts and starts the file again unlike the ATS (Advanced Teletext

System—The original BBC 'B' Teletext system) which stored the correct pages and re-tried for the corrupted ones. The manual states that it takes up to 8 hours for downloading all the files.

Fast-text

Fast-text is a simple form of five option menu with four links to different pages and one menu option to the subject index. The first four function keys and the print key are used to select the options. Those of you familiar with the original Teletext for the Beebs will be expecting the linked pages to be instantly available having been grabbed in advance by the software. But as far as I can see, the SCML Universal Adaptor does not have the 20 or so page grabbers available on the Acorn Adaptor that allow several pages to be grabbed in one Teletext pass.

Summary

The system is very usable despite one or two drawbacks which are a function of the Archimedes rather than the Adaptor. For example, the page takes a noticeable amount of time to display due to the absence of the mode 7 Teletext chip in the Archimedes which necessitates the use of an 80k screen mode emulation for displaying Teletext screens. However, this is a minor problem that a new user would not really notice.

I like the software, which seems to be well written, especially the tuning routine, though there are some weird effects if you knock a cursor key causing the cursor to reappear and fly around the top of the screen. The system as it stands is **only** a terminal and does not support the *-commands of the ATS. This means that writing programs for the Adaptor that require access to Teletext pages may be difficult. I understand that there is software and a separate lead available should you wish to use Teletext on another computer such as the BBC, Master or Compact and that they are going to make it

compatible with other computers like the Atari ST and the Amstrad PC as well.

The documentation was "economical" to say the least, although it did provide just about enough information to run the package.

I would like to see a neater software package and an error re-trying downloader as well as provision for SWI or *-commands which would allow access to Teletext pages from within user programs. Solidisk (who apparently wrote the software) say this could be around in the future if there is sufficient demand.

The unit is made by SCML, Solidisk's sister company, and is available from SCML at the special promotional price of £79.95. At this price it is a real bargain, but the offer only lasts until March 31st because SCML normally only deal with trade customers and not with the general public. After 31st March, you would have to get it from a dealer and not from SCML directly but it would be still well worth it at the normal price of £103. **A**

Readers' Survey

We're thinking of doing a readers' survey to find out what you think of the Archimedes; whether you've actually bought one (I know many of our readers are waiting before taking the plunge); if you've got one, what you use it for; what problems you've had and so on. If you can think of things that it would be useful to know about Archive readers let us know soon so that we can get it ready for next issue.

Nigel Stuart of Fairhurst Instruments has very kindly donated 5 copies of Arctist so we hope to use them as incentive to get people to fill in the questionnaire! **A**

NewsMaster Review

Steve Bruntlett

NewsMaster is an 'affordable desktop publishing' package from LTS Ltd. It's a two disc package, running under the Acorn PC Emulator, but only on an Archimedes 310! It allows you to create illustrated, multiple column A4 documents. The package is very easy to use and promises documents of your own, ready to print, 'in less than an hour'. All controls are via the function keys, whose legends are displayed on screen, and which are supported by single key-stroke on-screen help for all functions.

NewsMaster can be configured for single or dual drives as well as for a hard disk.

Although only a third of the newsletter can be seen life-size on the screen at any one time, NewsMaster displays screens that show full-page, full-width, normal and 'zoom' views, with eight levels of enlargement. Text, pictures and headlines can be edited and re-sized instantly. The style and size of the text, on screen, can be changed until you are happy with the effect.

The page layout can be changed to include up to ten columns with an optional headline block.

There are nine individual typefaces plus two typefaces available in up to 12 different type sizes. The typefaces are loaded from disc every time you edit the newsletter, so the screen can take a while to plot if it contains a lot of text. ASCII text files in IBM PC format can be loaded and used to flow round the page in a variety of different ways. NewsMaster supports over a hundred different printers, including dot-matrix, ink-jet and laser printers!

There are also sixty-three GLYPHS! (icons) in a variety of point type sizes, which are accessed from the keyboard.

The handbook supplied is very clear. It contains a stage by stage tutorial to start you off and also includes an extensive reference section for the artwork and text editing features. There are extensive cut and paste facilities in the text editing section. The artwork editing section includes re-sizing, mirroring, rotation and flipping of the supplied pictures. It also includes simple box facilities and texture infills.

As far as I know, you cannot load in your own images, (and this is the only limitation as far as I can see), but there is a library of one hundred small pictures supplied on disc to liven up your newsletters. The subjects of the images include graphics, business, fancy typeface, sports and travel. They can be enlarged to any shape or size.

I found NewsMaster to be a very user-friendly package and would have no hesitation in recommending its purchase to anyone who wants to produce simple newsletters. In terms of desk top publishing it is limited by the restriction of having to use the supplied images. The other limitation is the fact that you need an Acorn PC Emulator to run the package.

NewsMaster costs £60 + VAT **A**

More about Arctist

Nigel Stuart of Fairhurst Instruments, having read the review of Arctist asked if we would point out that the reviewer had "a very early pre-production copy" and add one or two comments.

The program only displays the colours once in the colour option bar and you can select a colour from within the picture itself. A lot more options are now offered on the function keys. The disc now contains demos of how to use the sprites. The manual has been completely re-written and the English corrected. **A**

**A VERY SPECIAL OFFER!
Spring 1988**

THE SCML TELETEXT ADAPTER:



**COMPATIBLE WITH ALL
BBC 'B's, MASTER 128K,
COMPACT & ARCHIMEDES**

£79.95

Including p&p and VAT
Cash on delivery at no extra
charge.
Money back guarantee if not
fully satisfied.

**OFFER MUST END
BY 31ST MARCH 88!**
When our normal price £103
inc. delivery must apply

SPECIFICATIONS:

- connects to the user port on Master 128k and BBC.
- connects to the joystick port on the Compact.
- connects to the Centronics port on the Archimedes.
- downloads all telesoftware, manual or automatic.

SPECIAL FEATURES:

- Mains powered. ON/OFF switch.
- Also works on other coputers such as ST, A500, Amstrad PC's (new lead and disc required).
- TV and sound outputs for Philips monitors.
- stylish two tone (brown and beige) metal case.
- complete with lead, software disc and manual.

★ PHONE SOUTHEND (0702) 335511 FOR IMMEDIATE DELIVERIES ★

Orders welcome from credit card holders (Access/Visa), trade and official bodies. Deliveries are normally ex-stock, but cheques require 7 day clearance. Please specify your computer.

MAIL ORDER:

Complete and return to:

TELETEXT SPECIAL OFFER SCML

87 Bournemouth Park Road
Southend-on-Sea
Essex SS2 5JJ

Please delete or tick as applicable.

Computer:

Name:

Address:

1) I enclose cheque/PO/Draft or debit my Access/Visa account:
for the amount of £79.95

2) or alternatively, I will pay cash on delivery at my local post office (£79.95).

3) send me data sheets on the TELETEXT.

I understand that if I am not entirely satisfied with your product, I am entitled to a full refund if I return the item in its original packaging within 7 days.

Signature: Date:

Archimedes 5.25 Disk Interface.

Fit a 5.25 Drive to your Machine the safe and easy way using our
NEW DISK INTERFACE

- ◆ Available Now
- ◆ Fully Buffered
- ◆ No Soldering
- ◆ Easy to Fit
- ◆ Supports 4 Drives
- ◆ All Cables Supplied
- ◆ Low Power Consumption
- ◆ Full Instructions

In Stock Now and Despatched the same day on receipt of your
Cheque / Postal Order for £24-00 inc P&P.

Dudley Micro Services

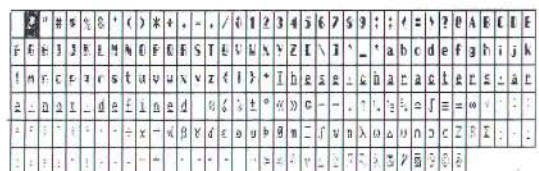
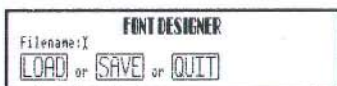
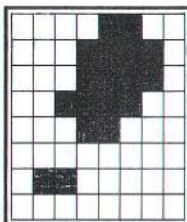
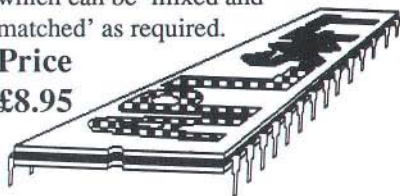
30 Hadley Close, Netherton, Dudley, West Mids. DY2 9JX

Tel: (0384) 633142

Font Designer

Design your own 8x8 BBC-fonts for the Archimedes. Fully mouse driven software. Price includes comprehensive manual and 15 different sample fonts, including scientific symbols, all of which can be 'mixed and matched' as required.

**Price
£8.95**



To: APL Software, 7 Ascendale, Deeping St James, Peterborough, PE6 8NZ.
Please supply one copy of **Font Designer**. I enclose a cheque/postal order for £8.95

Name:

Address:

.....

Fast Screenload / Screensave Module

Frank Dart, Michael Shaw and Jon Warmisham

As promised last month, here is our module for fast screenload and fast screensave. It allows you to *FastSave <filename> the entire screen in any graphics mode and the file it produces is readable by the normal *Screenload command or by the *FastLoad <filename> command in the same module.

Type in the listing and save it to disc (or buy the program disc!), then run the program and it will assemble the module and save it in the current disc directory as "FastRm". Then you can *RMLoad FastRm when you need to use it.

You can also *SET Alias\$ScreenLoad Fastload so that when other applications try to use the *Screenload command, the fast screenload is used instead.

Thanks to Michael, Jon and Frank for writing the module and also to Dave Clare of Clare's Micro for permission to publish this module since it will form part of their Advanced Toolkit.

```
10 REM > FastSrc
20 REM FastLoadSave
40 REM Michael Shaw, Jon Warmisham & Frank Dart
50 REM Subject to copyright, reprinted in Archive magazine only.
60
70 Module$      = "FastRm"      : REM module name
80 ModHandReason_Claim = &06     : REM action code to claim workspace
90 Module_ERRChunkNumber = &0200 : REM error number for module
100 Module_WorkSpaceSize= &0300 : REM module workspace size
110 FreeSpace      = &08        : REM byte offset to next free word
120 NextSprite     = &0C        : REM offset to next sprite
130 Width          = &1C        : REM width in words-1
140 ScanLines      = &20        : REM scan lines-1
150 SpriteOffset   = &2C        : REM offset from start of file of sprite
160 MaskOffset     = &30        : REM offset from start of file of mask
170 DefinedMode    = &34        : REM mode screen defined in
180 Storage        = &0280      : REM offset to general workspace
190 ModeFlags      = &02C0      : REM vdu variables modeflags
200 NColour        = &02C4      : REM max. logical colours 1, 3, 15 or 63
210 YBigFactor     = &02C8      : REM vertical pixel resolution
220 LineLength     = &02CC      : REM number of bytes on a pixel row
230 ScreenSize     = &02D0      : REM size of one screen buffer
240 DisplayStart   = &02D4      : REM start of display
250 Pointer        = &08        : REM pointer to parameter tail (filename)
260 Handle         = &08        : REM handle of filename
270 Mode           = &09        : REM mode screen defined in
280 Current        = &0A        : REM current screen mode
290 Stack          = &0B        : REM original stack pointer value
```

Fast Screenload/Screensave

```

300 PC          = &0F      : REM program counter register declaration
310 XOS_WriteI  = &20100: REM write character with V bit set
320 V_flag      = 1<<28 : REM V flag in PSR (&10000000)
330 DIM Code% 2048      : REM code offset word aligned
340 Code%=( (Code%+4)>>2)<<2
350 FOR Pass%=0 TO 2 STEP 2
360 P%=Code%
370 [OPT Pass%
380.Module_Base EQU D 0      ;module header
390              EQU D Initialise-Module_Base
400              EQU D 0
410              EQU D 0
420              EQU D Module_Name-Module_Base
430              EQU D Title_String-Module_Base
440              EQU D Module_Keywords-Module_Base
450              EQU D 0
460              EQU D 0
470              EQU D 0
480              EQU D 0
490.Module_Name EQU S "FastLoadSave"      ;module name
500              EQU B 0
510.Title_String EQU S "FastLoadSave"      ;module help text
520              EQU B 9
530              EQU S "2.01 (24 Jan 1988, "
540              EQU S "M.P.Shaw and J.Warmisham and F.Dart)"
550              EQU B 0
560              ALIGN
570.Initialise   STMTD R13!,{R7-R11,R14} ;initialise module
580              MOV  R0,#ModHandReason_Claim ;action code for claim
590              MOV  R3,#Module_WorkSpaceSize ;workspace size
600              SWI  "XOS_Module"          ;try to claim workspace
610              ADRVS R0,WorkspaceError ;workspace unavailable
620              STRVS R0,[R13]              ;so exit with error
630              LDMVSFD R13!,{R7-R11,R14}
640              ORRVSS PC,R14,#V_flag
650              STR  R2,[R12]              ;store workspace pointer
660              SWI  "XOS_Writes"          ;module initialise title
670              EQU S "FastLoadSave v2.01 (24 Jan 1988) "
680              EQUW &0D0A
690              EQU B 0
700              ALIGN
710              LDMFD R13!,{R7-R11,PC} ;exit module (initialised)
720.WorkSpaceError EQU D Module_ERRChunkNumber ;error text for workspace

```

```

730      EQU$ "FastRm unable to claim required workspace"
740      EQU$ 0
750      ALIGN
760.Module_Keywords EQU$ "FastLoad"          ;command table followed
770      EQU$ 0                               ;by help and syntax text
780      ALIGN
790      EQU$ FastLoad-Module_Base
800      EQU$ &00010001
810      EQU$ Load_syntax-Module_Base
820      EQU$ Load_help-Module_Base
830      EQU$ "FastSave"
840      EQU$ 0
850      ALIGN
860      EQU$ FastSave-Module_Base
870      EQU$ &00010001
880      EQU$ Save_syntax-Module_Base
890      EQU$ Save_help-Module_Base
900      EQU$ 0
910.Load_help EQU$ "**FastLoad allows the user to load screens "
920      EQU$ "much faster than the mos *ScreenLoad, but "
930      EQU$ "still allows total compatability with the "
940      EQU$ "mos and the sprite system."
950      EQU$ &0D0A
960.Load_syntax EQU$ "Syntax: *FastLoad <filename>"
970      EQU$ 0
980.Save_help EQU$ "**FastSave saves graphics screens to disc "
990      EQU$ "much faster than the mos *ScreenSave, but "
1000     EQU$ "still allows total compatability with the "
1010     EQU$ "mos and the sprite system."
1020     EQU$ &0D0A
1030.Save_syntax EQU$ "Syntax: *FastSave <filename>"
1040     EQU$ 0
1050     ALIGN
1060.Bad_file BL Module_error ;'File not found'
1070     EQU$ Module_ERRChunkNumber+1
1080     EQU$ "File not found"
1090     EQU$ 0
1100     ALIGN
1110.Bad_graphics BL Module_error ;'Not a graphics mode'
1120.Bad_graphics2 EQU$ Module_ERRChunkNumber+2
1130     EQU$ "Not a graphics mode"
1140     EQU$ 0
1150     ALIGN

```

Fast Screenload/Screensave

```

1160.Bad_monitor EQU D Module_ERRChunkNumber+3 ;'Non-existent mode'
1170 EQU S "Non-existent mode (wrong monitor type possibly)"
1180 EQU B 0
1190 ALIGN
1200.Bad_memory EQU D Module_ERRChunkNumber+4 ;'Not enough memory'
1210 EQU S "Not enough memory for this mode"
1220 EQU B 0
1230 ALIGN
1240.Module_error BIC R0,R14,&FC000003 ;link register points
1250 B System_error ;to error text
1260.CBad_mode CMN R0,#1 ;is it -1
1270 ADREQ R0,Bad_monitor ;yes, bad monitor error
1280 ADRNE R0,Bad_memory ;no, bad memory error
1290 B Close_error ;close file and flag err
1300.CBad_graphics ADR R0,Bad_graphics2 ;bad graphics error
1310.Close_error STMF D R13!,{R0} ;save error pointer
1320 MOV R0,#0 ;close file
1330 MOV R1,Handle
1340 SWI "XOS_Find"
1350 LDMFD R13!,{R0} ;recover error pointer
1360.System_error MOV R13,Stack ;recover stack pointer
1370 SWI "XOS_RestoreCursors" ;recover cursors
1380 STR R0,[R13] ;save error pointer
1390 LDMFD R13!,{R0-R12,R14} ;recover registers
1400 ORRS PC,R14,#V_flag ;exit signaling error
1410.SetUp_FastRm LDR R12,[R12] ;get workspace pointer
1420 STR R14,[R12,#Storage] ;save return address
1430 MOV Stack,R13 ;save stack pointer
1440 SWI "XOS_RemoveCursors";remove cursors
1450 MOV Pointer,R0 ;save parameter pointer
1460 BL ReadVariables ;read all mode variables
1470 LDR PC,[R12,#Storage] ;recover return address
1480.ReadVariables STMF D R13!,{R14} ;save return address
1490 ADR R0,ReadVars ;address of input block
1500 ADD R1,R12,#ModeFlags ;address of output block
1510 SWI "XOS_ReadVduVariables" ;read vdu/mode variables
1520 LDRB R0,[R12,#NColour] ;read logical colours
1530 ADD R0,R0,#1 ;colours=colours+1
1540 MOV R0,R0,LSL #3 ;colours=colours*8
1550 STR R0,[R12,#NColour] ;restore logical colours
1560 MOV R0,#&87 ;read current mode
1570 SWI "XOS_Byte"
1580 MOV Current,R2 ;save current mode number

```

```

1590      LDMFD R13!,{PC}           ;pull return address
1600.ReadVars  EQU 0      ;'ModeFlags'      - bit 0=1 non-graphics mode
1610      EQU 3      ;'NColour'      - max. logical colour
1620      EQU 5      ;'YBigFactor'    - vertical pixel resolution
1630      EQU 6      ;'LineLength'    - number bytes on pixel row
1640      EQU 7      ;'ScreenSize'    - size of one screen buffer
1650      EQU 149 ;'DisplayStart' - start of vdu display
1660      EQU -1 ;end of input block
1670.FastLoad  STMFD R13!,{R0-R12,R14} ;save registers on stack
1680      BL      SetUp_FastRm      ;setup relevant regs.
1690      MOV     R0,#&40            ;openin 'filename'
1700      MOV     R1,Pointer1
1710      SWI     "XOS_Find"
1720      BVS     System_error      ;file error
1730      CMP     R0,#0             ;if 0 'File not found'
1740      BEQ     Bad_file
1750      MOV     Handle,R0          ;save handle
1760      MOV     R1,R0             ;read first &238 bytes
1770      MOV     R0,#&04           ;from start of file
1780      MOV     R2,R12            ;header+max. palette size
1790      MOV     R3,#&0238
1800      SWI     "XOS_GBPB"
1810      BVS     Close_error      ;quit on err (close file)
1820      LDR     R0,[R12,#DefinedMode] ;check defined mode valid
1830      SWI     "XOS_CheckModeValid"
1840      BCS     CBad_mode
1850      MOV     Mode,R0           ;check if defined mode is
1860      MOV     R1,#0            ;indeed a graphics mode
1870      SWI     "XOS_ReadModeVariable"
1880      TST     R2,#1
1890      BNE     CBad_graphics
1900      CMP     Mode,Current      ;already in defined mode?
1910      SWINE    XOS_WriteI+22    ;No switch into the
1920      MOVNE    R0,Mode          ;defined mode
1930      SWINE    "XOS_WriteC"
1940      BLNE    ReadVariables    ;and read variables again
1950      BL      ReadPalette      ;restore saved palette
1960      MOV     R2,#&03          ;read from specified pos.
1970.FastLoad2  MOV     R0,R2      ;action code
1980      MOV     R1,Handle        ;handle
1990      LDR     R2,[R12,#DisplayStart] ;display start
2000      LDR     R3,[R12,#ScreenSize]  ;size of screen
2010      LDR     R4,[R12,#SpriteOffset] ;offset from start

```

Fast Screenload/Screensave

```
2020      ADD    R4,R4,#&0C      ;add three words
2030      SWI    "XOS_GBPB"      ;perform OS_GBPB
2040      BVS    Close_error     ;quit and close file
2050      MOV    R0,#0           ;close file
2060      SWI    "XOS_Find"
2070      MOV    R13,Stack        ;recover stack pointer
2080      SWI    "XOS_RestoreCursors" ;recover cursors
2090      LDMFD   R13!,{R0-R12,PC} ;pull all regs and exit
2100.FastSave STMFD   R13!,{R0-R12,R14} ;save registers on stack
2110      BL     SetUp_FastRm     ;setup relavent regs.
2120      LDR    R0,[R12,#ModeFlags];is current mode a
2130      TST    R0,#1           ;graphics mode
2140      BNE    Bad_graphics     ;non-graphics mode
2150      MOV    R0,#&0B         ;create empty sprite file
2160      MOV    R1,Pointer
2170      LDR    R2,FileType
2180      MOV    R3,#0
2190      MOV    R4,#0
2200      SWI    "XOS_File"
2210      BVS    System_error     ;quit on err (close file)
2220      MOV    R0,#&80         ;openin file
2230      SWI    "XOS_Find"
2240      BVS    System_error
2250      MOV    Handle,R0        ;save handle
2260      STMFD   R13!,{Handle-Stack};save registers
2270      ADR    R0,HeaderData    ;point to data
2280      LDMIA   R0,{R1-R11}     ;read twelve words
2290      MOV    R0,R12           ;point to workspace
2300      STMIA   R0,{R1-R11}     ;save twelve words
2310      LDMFD   R13!,{Handle-Stack};recover registers
2320      LDR    R0,[R12,#NColour] ;get palette size
2330      ADD    R0,R0,#&2C       ;add header size
2340      STR    R0,[R12,#SpriteOffset] ;offset to screen data
2350      STR    R0,[R12,#MaskOffset] ;mask offset (same)
2360      LDR    R1,[R12,#ScreenSize] ;size of screen to save
2370      ADD    R0,R0,R1         ;add palette size+header
2380      STR    R0,[R12,#NextSprite] ;offset to next sprite
2390      ADD    R0,R0,#&10       ;add 16 bytes
2400      STR    R0,[R12,#FreeSpace];offset to next word
2410      LDR    R0,[R12,#LineLength] ;bytes on pixel row
2420      MOV    R0,R0,LSR #2     ;words=bytes/4
2430      SUB    R0,R0,#1         ;words=words-1
2440      STR    R0,[R12,#Width]  ;save words-1
```

```

2450      MOV     R0,#1024                ;vertical pixel resolution
2460      LDRB    R1,[R12,#YEigFactor]    ;shift right value
2470      MOV     R0,R0,LSR R1           ;shift right by R1
2480      SUB     R0,R0,#1                ;subtract one
2490      STR     R0,[R12,#ScanLines];scan lines in mode
2500      STR     Current,[R12,#DefinedMode] ;store defined mode
2510      BL      WritePalette            ;save palette to workspace
2520      MOV     R0,#&02                  ;write &0238 bytes to file
2530      MOV     R1,Handle                ;which includes the
2540      MOV     R2,R12                   ;header data and the
2550      MOV     R3,#&0238                 ;maximum palette size
2560      SWI     "XOS_GBPB"
2570      BVS     Close_error              ;quit on err (close file)
2580      MOV     R2,#&01                  ;save data from pos.
2590      B       FastLoad2                ;use general routine
2600.HeaderData EQU    &00000001          ;number of sprites
2610      EQU    &00000010                ;byte offset
2620      EQU    &FFFFFFFF                 ;screen size+palette size+&1C
2630      EQU    &FFFFFFFF                 ;screen size+palette size+&0C
2640      EQU    "screendump"              ;sprite name (12 chars null)
2650      EQU    &0000
2660      EQU    &FFFFFFFF                 ;width in words-1
2670      EQU    &FFFFFFFF                 ;height in scan lines-1
2680      EQU    &00000000                ;first bit used (l. end of row)
2690      EQU    &0000001F                ;last bit used (r. end of row)
2700.FileType EQU    &00000FF9            ;sprite file type
2710.WritePalette STMF    R13!,{R0-R6,R14} ;save registers
2720      MOV     R4,#0                    ;logical colour
2730      ADD     R5,R12,#&38               ;address of palette data
2740      LDR     R6,[R12,#NColour]         ;colours used*8
2750.WritePalette2 MOV    R0,R4            ;logical colour
2760      MOV     R1,#&10                   ;normal colours
2770      SWI     "XOS_ReadPalette"         ;read palette
2780      STR     R2,[R5],#4               ;save first colour
2790      STR     R3,[R5],#4               ;save second colour
2800      ADD     R4,R4,#1                  ;next logical colour
2810      CMP     R4,R6,LSR #3              ;last colour?
2820      BNE     WritePalette2             ;repeat until done
2830      LDMFD   R13!,{R0-R6,PC}           ;return to caller
2840.ReadPalette STMF    R13!,{R0-R4,R14} ;save registers
2850      MOV     R2,#0                    ;logical colour
2860      ADD     R3,R12,#&38               ;address of palette data
2870      LDR     R4,[R12,#NColour]         ;colours used*8

```

```

2880.ReadPalette2 MOV    R0,R2,LSR #1      ;logical colour/2
2890                STRB   R0,[R12,#Storage+3];save logical colour
2900                LDR    R0,[R3],#4       ;get 1st or 2nd colour
2910                STR    R0,[R12,#Storage+4];save into workspace
2920                MOV    R0,#&0C         ;write palette action code
2930                ADD    R1,R12,#Storage  ;pointer to param block
2940                ADD    R1,R1,#3         ;add 3 to ptr
2950                SWI     "XOS_Word"       ;perform palette change
2960                ADD    R2,R2,#1         ;next logical colour
2970                CMP    R2,R4,LSR #2     ;last colour?
2980                BCC     ReadPalette2     ;repeat until done
2990                LDMFD  R13!,{R0-R4,PC}  ;return to caller
3000 ]
3010 NEXT Pass%
3020 OSLI ("Save "+Module$+" "+STR$(Code%)+ " "+STR$(P%))
3030 OSLI ("SetType "+Module$+" &FFA")
3040 OSLI ("RmLoad "+Module$) A

```

The Archimedes Hard Disk System

Alex de Vries

This article attempts to reveal some of the secrets behind interfacing a hard disk to the Archimedes computer. Please note that some of the information contained herein is speculative and no attempt is made to describe exactly how to interface a hard disk to the Archimedes. Clearly, no responsibility can be accepted for any damage resulting from the use of this information. Views on hard disk interfacing to the Archimedes are welcomed, as is constructive criticism.

Hard disks – the background

A hard disk comprises several parts. Obviously there is an actual hard disk. This unit is controlled by a controller. The controller has a bus on it called SASI or SCSI (later version). This controller is what the computer talks to. All it has to do is ask for certain tracks/sectors and the controller will do the rest. Unfortunately this is not the end of the matter since the SCSI bus is a large

bus – larger than 8 bits, anyway, which is the breakpoint for most systems – and furthermore it operates with negative logic. The problem is solved by using a host controller which inverts the logic and provides for byte-sized registers. Please note that the hard disk system has nothing to do with the floppy interfacing or software.

Archimedes hard disks

Connecting a hard disk to the Archimedes is basically an operation of purchasing the Acorn Hard disk Podule. This podule is unfortunately expensive (*and not actually available yet! Acorn have had my money since October! Ed.*) – at £499 plus VAT for a 20 Mbyte drive it is grossly over-priced. (A similar system can be purchased for an IBM computer for about £200 plus VAT.) The price Acorn charges should buy about 30 to 40 Mbyte (quoted prices include a controller).

So, what is so special? Frankly nothing, except that nobody outside Acorn seems to know (yet) how to interface a hard disk to the Archimedes, not for lack

of hardware knowledge, but for a lack of software information. The Reference Manual is extremely short on this matter, giving only vague information. We can only hope that in the near future, someone can tell us exactly how Acorn have designed their system.

BBC Compatibility

Some very worried people must be wondering if their (just as expensive) BBC hard disk can be fitted to the Archimedes. In short, the answer is no. At least, that is what Acorn will tell you – it is even stated explicitly at the beginning of the I/O podule manual.

So what is the reason? Basically the hardware is OK but the software isn't. This knowledge must lead us to at least one conclusion: Acorn have not used the same protocol as with the BBC. This may seem a silly conclusion, but it is important. Acorn have included a call in the ADFS which allows the user to specify where the hard disk (physically) is to be found. So if they allowed for such cases, they must have anticipated various controllers. So why not implement code that allows existing systems to be used?

Another piece of interesting information can be gained from an inspection of the circuit board. Anybody who has seen a hard disk interface will be amazed at the simplicity of the Acorn interface. One single (large) IC seems to be doing all the work. The IC is an Hitachi device (HD 63463) but that is about the limit of our current knowledge. Two questions arise from this information:

1. Is this IC a fantastic new device incorporating Hard disk controller and some host interface? In that case is it SCSI?
2. Is Acorn cutting corners and have they used some trickery to get the system going (like leaving out the host interface or using negative logic)?

Armed with this information we could deduce:

- (a) Since the BBC interface is SCSI standard and uses a well defined and usable host interface, can we assume the Archimedes does not contain (compatible) SCSI code?
- (b) Acorn have left out the host interface and are talking directly to a SCSI bus. (considered unlikely due to the apparent lack of latches on the PCB needed to implement such a system)
- (c) The controller chip has a different control structure and is not SCSI compatible. (considered unlikely, since SCSI is really the standard, or is there a new standard out?)
- (d) The controller has a host interface (negative logic or not) and is wildly different from the BBC host adapter.

We have a number of options available, as shown. But are there any more?

Conclusions

In theory, it should be perfectly possible to use a BBC hard disk on the Archimedes, but:

- code has to be written to drive it (not too difficult as only a few routines are required)
- an I/O podule is required for the 1 MHz bus which adds to the overall cost.
- the hard disk code would have to be loaded from floppy before the hard disk is usable, unless a ROM is programmed (or it goes in battery-backed ram) and placed onto a ROM podule which means more cost.

References

- Winchester Disk Systems M.J.Tubby 1987
 Seagate SCSI interface manual Seagate 1986
 (RS part no 635-779)
 Podule Application Note Acorn 1987

Some interesting additional facts

- A Fuji 3.5" hard disk 20 Mbyte costs £195 plus VAT (retail)
- A SCSI and host interface costs £100 to £150
- Acorn have reserved SWIs and IDs for a SCSI interface **A**

The 5-byte Time Format

Tony Cheal

In response to our HELP call in January Archive (page 6), Tony Cheal (winner of our first and only programming competition) has written us a routine that converts the time into the 5 byte format used by the OS for things like date-stamping files. This routine is the reverse, therefore of the "OS_ConvertStandardDateAndTime" routine. The program below illustrates and tests its use. If you type in a suitable time\$ (a sample one is given to you by printing the current time\$ on the screen for you to copy) the program converts it into 5 byte format and then uses the OS routine to convert it back again just to check that it is correct.

The version PROC5B works for dates up to 18th Mar 2074 but overflow stops it working for later dates. PROC5B2 will work in the full range to 3rd Jun 2248. You will recognise bits from the day-converter routine. The year is made relative to 1900 with the months starting at March to simplify matters. There are 58+1 (sic) days before March 1900. There are 33750*256 centiseconds in a day (fortunately divisible!). There are the usual bits for leap years: + Y%DIV4 for each leap year, - Y%DIV100 for centuries, + Y%DIV400 for leap centuries, and a snazzy bit for the months (any bad month=Mar). You don't have to have the weekday right.

The PROC5B version can be paraphrased: LOCAL year: Clear top byte : year=<year> + adjustment for Jan/Feb - 1900 Centiseconds=((<hours>*60+<minutes>)*60)+<seconds>)*100 Lots-of-256-centiseconds+=(year*365+<number-of-days-from-start-of-year-to-first-of-month(!)>+<leap-days>+<day-in-month>)*33750

```

10 REM>5BYTE
30 DIM B%5,X%255
40 PRINT TIME$
50 REPEAT INPUTLINE""T$
60   PROC5B(T$,B%)
70   FOR I%=0TO4
80     PRINT~B%?I%;
90   NEXT
100  PRINT
110  SYS"OS_ConvertStandardDateAnd
    Time",B%,X%,100 TO,Y%,L%:?Y%=&D
120  PRINT$X%
130  PRINT
140  UNTILO
150  END
160
170 DEFPROC5B(T$,B%)LOCALY%:B%?4=0
    :Y%=VALMID$(T$,12,4)+(INSTR("aneb",
    MID$(T$,9,2))>0)-1900:!B%=( (VALMID$
    (T$,17,2)*60+VALMID$(T$,20,2))*60+
    VALMID$(T$,23))*100+.5:B%!1+=(Y%*365
    +(153*INSTR("xprayunulugepctovecaneb"
    ,MID$(T$,9,2))+4)DIV10+58+Y%DIV4-
    Y%DIV100+(Y%+300)DIV400
    +VALMID$(T$,5,2))*33750:ENDPROC
180
190 DEFPROC5B2(T$,B%)LOCALY%:B%!2=0:
    Y%=VALMID$(T$,12,4)+(INSTR("aneb",
    MID$(T$,9,2))>0)-1900:!B%=( (VALMID$
    (T$,17,2)*60+VALMID$(T$,20,2))*60+
    VALMID$(T$,23))*100+.5:B%!1+=
    (Y%MOD256*365+(153*INSTR(
    "xprayunulugepctovecaneb"
    ,MID$(T$,9,2))+4)DIV10+58+Y%DIV4-Y%
    DIV100+(Y%+300)DIV400+VALMID$
    (T$,5,2))*33750
200 B%!2+=Y%DIV256*12318750
210 ENDPROC
```

PINEAPPLE SOFTWARE

DIAGRAM II for the ARCHIMEDES

Diagram II represents a major breakthrough in the techniques used for drawing software on the BBC micro. It works on a completely different principle to other drawing software by storing the drawing information straight to disc as coded 8x8 pixel blocks.

This technique has three major advantages. First, the size of the diagram is only limited by the amount of space on disc. Second, there is no limit to the amount of information which can be stored on a given diagram as the information is stored on disc and not in computer memory. Third, the disc storage technique allows the smooth scrolling of the screen over the whole surface of the diagram. Up to about 120 Mode 0 screens on an 800k Archimedes disc.

For people not familiar with Diagram, the basic operation of the program is to first display any part of the diagram, either by quoting an index name or screen number. At this point you are free to scroll the diagram around the screen using either cursor keys, or Archimedes mouse. You may stop scrolling at any point and enter the edit mode which allows you to modify or add new information to the diagram. When you have finished making alterations, the new information is stored to disc and you are free to scroll elsewhere on the diagram stopping to edit at any time.

The edit features are now very comprehensive. Text may be typed straight from the keyboard, and if other fonts are stored as user defined characters then these fonts may be printed using the normal keyboard keys.

The print routines are now completely 'scaleable' with both the horizontal and vertical scales being totally variable in 1% steps between a size that will give 18 mode 0 screens on an A4 sheet and a size that would print just one pixel on a sheet! The routines are adaptable to most types of dot matrix printer and full advantage may be made of wide carriage printers. The printouts may be rotated through 90 deg. if required.

The new editing features make DIAGRAM II suitable for all types of serious drawing application including scale drawings, flow charts, architectural, family trees, and many other subjects as well as circuit and schematic diagrams. As an example of what Diagram II can do, this complete advertisement has been produced with the package.

1. Provides up to 880 User Definable Characters stored in memory. Rapid horizontal and vertical line drawing routine with automatic joins for circuit diagrams.
2. Full rubber band line drawing and circle drawing modes.
3. Takes advantage of the Extended graphics facilities of the Archimedes to provide drawing of arcs, sectors, chords, parallelograms, ellipses and flood filling of areas.
4. Pixel cursor drawing and deleting mode allows very fine detail to be added.
5. Defined areas of screen may be moved, copied, deleted or saved to disc effectively allowing unlimited numbers of icons up to a full screen size each.
6. On-screen indication of cursor position shows either the cursor position on the overall diagram or the distance from a preset point.
7. Keyboard keys may be predefined to print User Defined Characters, enabling new character sets to be stored as user defined characters and then printed straight to the screen using the keyboard keys.
8. Wordprocessor files may be loaded to the screen and automatically formatted into any shape screen area.
9. Index names may be set up to point to given areas of a large diagram, to enable rapid access to a given point on a diagram.
10. Fully compatible with the Archimedes Mouse.
11. All Diagram 'Utilities' are included, enabling screen ident numbers and borders to be added, and any area of diagram to be moved or copied. The whole diagram may be displayed in reduced scale (either 4x4 or 8x8 format) on a single screen, and the size of the overall diagram may be increased or decreased.
12. Completely scaleable print routines allow any area of the diagram to be printed either horizontally or through 90 deg. in scales that may be varied in 1% steps allowing up to 18 mode 0 screens to be printed on an A4 sheet (still with readable text).
13. Complete with 40 page easy to understand handbook.



Diagram II runs on the Archimedes 6502 emulator but at about 4 times the speed of the BBC version. It is supplied on 3.5" 800k disc.

DIAGRAM II - £55.00 + vat

P & P free (except overseas)

N.B. Upgrades are available to existing Diagram owners. Please contact Pineapple if you have not received an upgrade letter.

VISA

39 Brownlea Gardens, Seven Kings, Ilford, Essex IG3 9NL Tel 01-599 1476

Access

The WIMP Environment – Part 4

Adrian Look

As promised last month, I shall now look at how to input text in the WIMP environment. The WIMP environment is very fussy. It likes everything done 'just-so' or else it gets confused and produces undesirable effects.

In order to bring about 'legal' text input, we will have to cause the 'Wimp_Poll' routine to react to two new conditions:

Mouse_click (6) – we now want the AWM to report to us if the user clicks a mouse button while the pointer is in a window. This will allow us to put the text cursor (caret) at the cursor position in the relevant window. For this condition to be generated by the AWM, the window in question must have a 'button type'. The 'button type' is stored at block%!60, when we define a window (cf Vol 1, No 2 & 3). The button type we require is 2. Thus we must set block%!60=&2000. In a more general case, button type 'n' requires block%!60=&n000.

Key_pressed (8) – once the caret has been placed in the window (i.e. the window has an 'input-focus'), the AWM will generate this condition every time a key is pressed.

Thus we get a new 'Wimp_Poll' routine:

```
REPEAT
SYS "Wimp_Poll",0,block% TO
reason
CASE reason OF
:
WHEN 6 : PROCmouse_click
:
WHEN 8 : PROCkey_pressed
:
ENDCASE
UNTIL FALSE
END
```

Now, we need to put the caret in the relevant window when the mouse button is clicked. The is done by using:

'Wimp_SetCaretPosition' at &400D2

This call needs the window handle, the co-ordinates of the caret in the window, and the caret size.

If we read the position of the mouse (in absolute terms), we can then find the caret position (in window values) using the relevant window's absolute co-ordinates – found using 'Wimp_GetWindowInfo' (described in an earlier article). The mouse position is found by using 'Wimp_GetPointerInfo' (described later).

```
DIM block% &100
caret_size=8
:
DEFPROCmouse_click
SYS
"Wimp_GetPointerInfo",,block%
mx=block%!0 : REM mouse co-
ordinates
my=block%!4
handle=block%!12
block%!0=handle
SYS"Wimp_GetWindowInfo",,block%
x0=block%!4 : REM top left of
work-area
y1=block%!16
scx=block%!20 : REM position of
work-area in main area
scy=block%!24
cx=mx-x1+scx : REM caret co-
ordinates
cy=my-y1+scy
SYS "Wimp_SetCaretPosition",
handle,-1,cx,cy,&1000000
OR caret_size
ENDPROC
```

Note that if you call 'Wimp_SetCaretPosition' with the handle set to -1 then the caret will be switched off.

Now that the window has been given the 'input focus', any keys that are pressed will be registered by the 'Wimp_Poll' routine. In order to find out which key had been pressed and where it should go:

```
DEFPROCkey_pressed
handle=!block% : REM window in
                  which caret is located
cx=block%!8 :REM caret position
               when key was pressed
cy=block%!12
key=block%!24 : REM the ASCII
               value of the key pressed
:
REM Process key press
:
ENDPROC
```

Having found this out, you are free to process the key press as you like – the best way is to store the text and cause a 'Window_Update', your own window drawing routine (see last month) accessing the stored text and displaying it.

Here are a few other Window Manager routines:

'Wimp_GetPointerInfo' at &400CF

This routine tells you all about the mouse pointer:

```
DIM block% &100
:
PROCpointer_info
:
END
:
DEFPROCpointer_info
SYS "Wimp_GetPointerInfo",,
                                block%
mx=block%!0
my=block%!4
```

```
mb=block%!8
handle=block%!12 : REM (if not
                  over window then -1
                  else window handle)
icon=block%!16 : REM (if not
                over icon then -1
                else icon handle)
ENDPROC
```

where mx = mouse x co-ordinate
 my = mouse y co-ordinate
 mb = button state (same result as for the BASIC command, MOUSE)
 bit 0 – ADJUST
 bit 1 – MENU
 bit 2 – SELECT

'Wimp_CloseDown' at &400DD

This routine only exists for WIMP version 0.18 onwards (that's why I haven't mentioned it earlier). Once you have finished with the WIMP environment you should issue this call. (Programmers who wrote the OS1.2 desktop, please note!!) The routine will then tidy everything up – for example, the function keys.

```
PROCclosedown
END
:
DEFPROCclosedown
SYS "Wimp_CloseDown"
ENDPROC A
```

That's all for this month. Thank you for all the feedback you have been sending. When trying to explain something as complicated as the Window Manager, it's important to let us know if you reckon we haven't explained something properly, so keep those letters coming.

Thank you, *Adrian*

The Game of Life

Dr Tony Brain

The game of Life was developed by Professor John Conway at the University of Cambridge and described in 'Scientific American' in 1970.

The game is played on a square grid with each square (or cell) being in one of two states – alive or dead. The state of each cell is determined by the states of each of its eight neighbours according to the following rules:–

- 1) Every live cell with either two or three neighbours survives to the next generation.
- 2) Every live cell with less than two or more than three neighbours dies (of loneliness or overcrowding).
- 3) Every dead cell with exactly three neighbours becomes alive in the next generation.

The rules are applied to every cell simultaneously to create a new generation. The player's role is just to set up the initial pattern. Although the rules are simple it is very difficult to predict the evolution of even a simple pattern. For example a square block of four cells remains static while a pattern of five cells called an R-pentomino will, given a large enough grid, continue to evolve for 1103 generations before becoming static. Therein lies the fascination of the game.

Examples of Life Patterns

Listed below are some of the categories of Life patterns and examples of each.

1) Still Life



Block Beehive Loaf Eater

2) Oscillators



Toad Clock Tumbler

3) Gliders and Spaceships



Glider Spaceship

4) 'Unlimited' Growth



R-pentomino Acorn

Other categories exist such as glider guns, puffer trains, breeders etc. but these are mostly too large to include here. Readers are referred to a book called 'The Recursive Universe' by William Poundstone for more information.

Program Notes

The program presented here uses an 80 x 60 grid and, for speed, uses machine code to calculate each new generation and plot to the screen. The display wraps round at the sides of the screen, anything going off the right side appearing one row down on the left and vice-versa. The Life rules break down at the top and bottom of the display e.g. a glider becomes a block when it hits the edge of the screen. Within these limitations it is still possible to explore a very large variety of Life objects.

On running the program six options are given: **Setup** allows you to enter a specific pattern; **Clear** clears the screen;

Random adds 50 random points to the existing display;

Stop calculates and displays generation;
Continues calculations if the user continues
generation until control is passed
Ends the program.

In line 130, SYS"OS_ReadVduVariables" with 149 in the parameter block reads the start address of screen memory. It is important to do this as the address varies with machine configuration.

In line 180, INKEY(-10) tests for the select button pushed. (INKEY(-11) and -12 for the menu and adjust buttons.)

In line 1410, *FX 21 9 flushes the mouse buffer.

References

'Mathematical Games' by Martin Gardner.
Scientific American October 1970

'The Recursive Universe' by William Poundstone. Oxford University Press. ISBN 0-19-285173-X

```

10 REM >Life
20 REM by Dr Tony Brain
30 MODE12
40 ON ERROR:PROCError:END
50 PROCinit
60 REPEAT
70   REPEAT
80     MOUSE x%,y%,b%
90     UNTIL y%<64 AND b%=4
100    IF x%<384 THEN PROCsetup
110    IF x%>384 AND x%<496 PROCclear
120    IF x%>496 AND x%<624 PROCrandom
130    IF x%>624 AND x%<720 PROClife
140    IF x%>720 AND x%<912 THEN
150      PROCflush
160      REPEAT
170        PROClife
180        UNTIL INKEY(-10)
190      ENDIF
200    IF x%>912 THEN stop%=TRUE
210    PROCflush

```

```

220 UNTIL stop%
230 MODE12
240 END
250 :
260 DEFPROCError
270 MODE 12
280 PRINT REPORT$+" at line ";ERL
290 ENDPROC
300 :
310 DEFPROCinit
320 DIM code% 7000,osblock% 16
330 osblock%!0=149
340 osblock%!4=-1
350 SYS"OS_ReadVduVariables",
    osblock%,osblock%+8
360 scrnaddr%=osblock%!8
370 PROCassemble
380 stop%=FALSE
390 COLOUR0,64,64,64:COLOUR1,0,240,
    176
400 COLOUR2,240,48,0:COLOUR15,240,
    240,240
410 VDU19,1,24,128,128,128
420 COLOUR15
430 PRINTTAB(1,30)"Generation";
    TAB(1,31)"Live Cells";
440 COLOUR1
450 PROCmenu1
460 GCOL3:VDU5:MOVE 1039,44
470 PRINT"Archive Life":VDU4
480 PROCclear
490 *POINTER 1
500 MOUSE ON
510 MOUSE RECTANGLE 271,0,732,64
520 OFF
530 ENDPROC
540 :
550 DEFPROCclear
560 generation%=1:number=0
570 FOR no%=0 TO 4960 STEP4
580   array%!(no%=0
590   NEXT
600 CALL display
610 PROCprint
620 ENDPROC
630 :
640 DEFPROCrandom
650 FOR no%=1 TO 50
660   ?(array%+485+RND(70)+(80*RND
    (50)))=1
670 NEXT

```

```

680 CALLdisplay
690 ENDPROC
700 :
710 DEFPROCsetup
720 LOCAL stop%
730 stop%=FALSE
740 MOUSE OFF
750 PROCflush
760 MOUSE RECTANGLE 0,64,1271,944
770 PROCmenu2
780 PROCbox:REPEAT
790   WAIT:PROCbox
800   MOUSE X%,Y%,b%
810   X%-=X% MOD 16:Y%-=Y% MOD 16
820   IF Y%<64 THEN Y%=64
830   WAIT:PROCbox
840   IF b%=1 THEN stop%=TRUE
850   IF b%=4 THEN PROCcell(1)
860   IF b%=2 THEN PROCcell(0)
870 UNTILstop%:PROCbox
880 MOUSE ON
890 MOUSE RECTANGLE 271,0,732,64
900 PROCmenu1
910 ENDPROC
920 :
930 DEFPROClife
940 CALL calculate
950 generation%+=1:PROCprint
960 ENDPROC
970 :
980 DEFPROCbox
990 GCOL4,0:MOVEX%-8,Y%-4
1000 PLOT1,24,0:PLOT1,0,24
1010 PLOT1,-24,0:PLOT1,0,-24
1020 ENDPROC
1030 :
1040 DEFPROCprint
1050 COLOUR1
1060 PRINTTAB(12,30);FNstr
      (generation%);TAB(12,31);
      FNstr(!number);
1070 ENDPROC
1080 :
1090 DEFPROCmenu1
1100 RESTORE 1220
1110 GCOL0:RECTANGLEFILL 271,0,732,64
1120 VDU5
1130 FORno%=1TO6
1140   READkey$,pos%,len%
1150   GCOL2:RECTANGLE FILL pos%,0,
      len%,64
1160   GCOL0:RECTANGLE FILL pos%+8,
      8,len%-16,48
1170   GCOL15:MOVEpos%+16,44
1180   PRINTkey$
1190 NEXT
1200 VDU4:OFF
1210 ENDPROC
1220 DATA Setup,271,108,Clear,383,108
1230 DATA Random,495,124,Step,623,92
1240 DATA Continuous,719,188,Exit,
      911,92
1250 :
1260 DEFPROCmenu2
1270 GCOL4:RECTANGLEFILL271,0,732,64
1280 GCOL0:RECTANGLEFILL279,8,716,48
1290 GCOL15:VDU5:MOVE295,44
1300 PRINT"Select - fill Menu -
      empty Adjust - end"
1310 VDU4:OFF
1320 ENDPROC
1330 :
1340 DEFPROCcell(c%)
1350 GCOLc%:?FNarraypos=c%
1360 RECTANGLE FILL X%,Y%+4,8,8
1370 ENDPROC
1380 :
1390 DEFPROCflush
1400 REPEATMOUSEX%,Y%,b%:UNTILb%=0
1410 *FX21 9
1420 ENDPROC
1430 :
1440 DEFFNstr(num%)
1450 =RIGHT$(" " +STR$(num%),4)
1460 :
1470 DEFFNarraypos
1480 =array%+(X%/16)+(64-Y%/16)*80
1490 :
1500 DEFPROCassemble
1510 rows%=60:columns%=80
1520 scrn_addr=0:pointer=1
1530 cell_value=2
1540 array_addr=3:array_end=4
1550 live_count=5:offset_value=6
1560 rows=7:columns=8
1570 loop_count=9:colour=10
1580 live_no=11:no_addr=12:sp=13
1590 FOR pass%=0TO2STEP2:P%=code%
1600
1610 [OPT pass%
1620 .offset
1630 EQU0 &50020100

```

```

1640 EQUED &A2A1A052
1650.calculate
1660 STMFD (sp)!, {R0-R12,R14}
1670 ADR array_addr,array%
1680 ADD array_end,array_addr,#2480
1690 ADD array_end,array_end,#2400
1700.loop
1710 ADR pointer,offset
1720 MOV live_count,#0
1730 MOV loop_count,#0
1740.count
1750 LDRB offset_value,[pointer,
                        loop_count]
1760 LDRB cell_value,[array_addr,
                        offset_value]
1770 MOVS cell_value,cell_value
                        ,LSR #1
1780 ADDCS live_count,live_count,#1
1790 ADD loop_count,loop_count,#1
1800 CMP loop_count,#8
1810 BLT count
1820 LDRB cell_value,[array_addr,
                        #81]
1830 ANDS cell_value,cell_value,#1
1840 BEQ live
1850 CMP live_count,#2
1860 BEQ mark
1870.live
1880 CMP live_count,#3
1890 BNE next
1900.mark
1910 ORR cell_value,cell_value,#2
1920 STRB cell_value,[array_addr,
                        #81]
1930.next
1940 ADD array_addr,array_addr,#1
1950 CMP array_addr,array_end
1960 BNE loop
1970 ADR array_addr,array%
1980.shift
1990 LDRB cell_value,[array_addr]
2000 MOVS cell_value,cell_value
                        ,LSR #1
2010 STRB cell_value,[array_addr]
                        ,#1
2020 CMP array_addr,array_end

2030 BNE shift
2040 B start
2050.display
2060.screenstart EQUED scrnaddr%
2070.number EQUED 0
2080.pixel EQUED &00111111
2090 STMFD (sp)!, {R0-R12,R14}
2100.start
2110 MOV live_no,#0
2120 ADR no_addr,number
2130 ADR array_addr,array%
2140 ADD array_addr,array_addr,#80
2150 ADR pointer,screenstart
2160 LDR scrn_addr,[pointer]
2170 ADR pointer,pixel
2180 MOV rows,#rows%
2190.yloop
2200 MOV columns,#columns%
2210.xloop
2220 MOV colour,#0
2230 LDRB cell_value,[array_addr]
                        ,#1
2240 CMP cell_value,#1
2250 ADDEQ live_no,live_no,#1
2260 LDREQ colour,[pointer]
2270 MOV loop_count,#3
2280.plot
2290 STR colour,[scrn_addr],#320
2300 SUBSloop_count,loop_count,#1
2310 BNE plot
2320 SUB scrn_addr,scrn_addr,#956
2330 SUBS columns,columns,#1
2340 BNE xloop
2350 ADD scrn_addr,scrn_addr,#960
2360 SUBS rows,rows,#1
2370 BGT yloop
2380 STR live_no,[no_addr]
2390 LDMFD (sp)!, {R0-R12,R15}
2400 :
2410.array%
2420 ]
2430 P%+=4960
2440 [.end%]
2450 NEXT pass%
2460 ENDPROC A

```

More about Pointers

Adrian Look

Last month, we saw Keith McAlpine's pointer designer, with Jeff Shipp's modified ARM code program to set up the definition. The pointer was set up using OS_Word &15. In this article, I shall discuss the principles behind setting up a mouse pointer using this call and in particular will show you how you can have four different pointers and switch between them.

'OS_Word' &15 (21) – define pointer and mouse parameters

This call defines a pointer's size, shape and active point. The 'active point' is the pixel on the pointer which actually does the pointing!

In order to do this, we have to set up a parameter block containing the following information:

Position:	Contents:	Range:
block?0	0	
block?1	Shape number	1 – 4
block?2	Width (w) in bytes (four pixels to a byte)	0 – 8
block?3	Height (h) in pixels	0 – 32
block?4	ActiveX, pixels from left	0 – w*4-1
block?5	ActiveY, pixels from top	0 – h-1
block?6	Least signif'nt byte of pointer to data	
block?7	.	.
block?8	.	.
block?9	Most signif'nt byte of pointer to data	

You will notice that you can define four pointer shape numbers. Having done this you can select the desired shape by using:

MOUSE ON shape_number

The OS_Word routine does not allow the individual shapes to store their colour definitions. So if you use pointers with different

colour definitions, you will have to set up the colours every time you switch between pointers.

The program below (written in BASIC for the benefit of those of you who have not used ARM code yet!) uses the output files (which are parameter blocks – as described above, plus colour definitions) from Keith McAlpine's designer program and allows you set up the pointer as any shape number.

```

10 REM > SetPointer
20 DIM block% &400
30 MODE 0
40 REM turn on the pointer before
    defining shapes
50 *POINTER
60 REPEAT
70   INPUT "Pointer filename:"
    filename$
80   INPUT "Shape number:" number
90
100  OSCLI("LOAD "+filename$+" "+
    STR$(block%))
110
120  block%?1=number
130  table=block%+19
140  block%?6=table
150  block%?7=table DIV &100
160  block%?8=table DIV &10000
170  block%?9=table DIV &1000000
180
190  col=block%+10
200  MOUSE COLOUR 1,col?0,col?1,
    col?2
210  MOUSE COLOUR 2,col?3,col?4,
    col?5
220  MOUSE COLOUR 3,col?6,col?7,
    col?8
230  SYS "OS_Word",&15,block%
240  MOUSE ON number
250
260  INPUT "Again?" A$
270  UNTIL A$="N" OR A$="n"
280  END

```

More about BASIC V

Dr Colin Dean

• Starting BASIC from in-core programs

It is possible, when entering BASIC, to tell the interpreter to start with a program that is already present somewhere in memory and, optionally, to RUN this program when starting. The program can be in one of 3 forms:

- a tokenised BASIC program
- the text of a BASIC program with line numbers
- BASIC text without line numbers

BASIC will automatically tokenise the last 2, and add line numbers if needed.

You need to tell BASIC the addresses in memory of the start of the program or text, and the first byte after its end. These must be given as 8-digit hex values, without stripping off leading zeros. *(It seems that it will work as long as the upper address limit is greater than the length of the file. Ed.)*

Simple Example:

We have a BASIC program on disc in the file "myprog", which is &144 bytes long, and we are at the operating system's '*' prompt:

```
*Load myprog 10000
*Basic -Chain @00010000,00010144
```

The '@' and ',' are crucial. BASIC automatically copies the program from wherever it is located to the current value of PAGE. If the '-Chain' keyword is omitted, the program is copied down to PAGE, but not RUN.

This facility is most likely to be useful if you want to RUN a BASIC program that's stored in a module or podule.

• Running BASIC programs as *-commands

You may be aware that BASIC programs can be run from disc as *-commands, provided their file-type (as displayed by *Info) is 'BASIC'.

Example:

```
*Myprog \Enter BASIC and CHAIN
\ program 'Myprog'
```

When the program ends (e.g. by END or STOP), BASIC does a QUIT. When a program is RUN in this way (in Arthur 1.20 / BASIC 1.02) the BASIC copyright message is not printed, so the user of the program need not even know that the program was written in BASIC.

What you may not have known is that keywords can be passed to the program,

Example:

```
*Myprog have a nice day
```

The following BASIC program demonstrates this. It uses the SYS call "OS_GetEnv" to find the address of the command string and then copies it to the string variable command\$.

```
10 REM > argdemo
20 REM demonstrate command
   arguments in BASIC
30 REM Colin C Dean 08-02-88
40 :
50 SYS "OS_GetEnv" TO pointer%
60 command$ = ""
70 offset% = 0
80 WHILE pointer%?offset% <> 0
90   command$ += CHR$
      (pointer%?offset%)
100  offset% += 1
110 ENDWHILE
120 PRINT command$
```

If you SAVE this to disc as 'argdemo', and then call it by:

```
*Argdemo This is some text
the program prints:
BASIC -quit "Argdemo" This
is some text
```

So to get the command options you must split command\$ after the 2nd quote.

• Faster Plotting in Arthur 1.20 / BASIC 1.02

One of the reasons why plotting is faster in the new chip set (see Archive issue 5) is that there is a new SWI call, "OS_Plot", which BASIC uses instead of VDU 25 for PLOT, DRAW, MOVE, CIRCLE etc.

You can demonstrate it yourself from BASIC, for example:

```
SYS "OS_Plot", 5, 250, 400
this has the same role as
VDU 25, 5, 250; 400;
or
MOVE 250, 400
```

There is, however, a bug in OS_Plot. Its code is intended to call the VDU drivers directly (hence the extra speed) provided output is currently going to the VDU only, but to revert to calling OS_WriteC (i.e. as VDU 25) if there is any other output (e.g. a *spool file or the RS423 port). However, a single-instruction bug means that it always uses the fast method and completely ignores *fx3 settings. As a result, you can't spool graphics to a file or to a peripheral device from BASIC unless you replace all relevant graphics statements by their VDU 25 equivalents.

There is a simple patch: the following poke solves the problem:

```
!&01F03510 = &0380CF14
```

but it reduces the speed of OS_Plot to that of VDU 25.

(So presumably you could do something like:

```
patch%=&01F03510
fast%=!patch%
slow%=&0380CF14
```

Then you could use !patch%=slow% when you want to spool a graphics file and !patch%=fast% when you want to speed things up again. Ed.)

Come and see us at the Show!

If you would like to come and talk to us or show us your latest piece of exciting software – or (hopefully) re-new your subscription if it was only a 6-month one, why not come to one of the Micro User Shows?

Manchester (UMIST) March 18th – 20th

London (Horticultural Hall) May 13th – 15th

At Manchester we'll be on stand number 1 and in London on stand number 2.

We'll try to have some discount hardware and software for you to buy through the magazine – members only!

Better still, how about coming to help us on the stand? Two or three people helped me at the last show in November and found it an enjoyable(?) experience. You don't have to be (or pretend to be!) an expert – just to be there and chat to folk – and take their money!! – that's all that's needed. If you could give me even an hour or two, that would be great – give me a ring and I'll send you a complementary ticket. **A**

The Floating Point Emulator / Co-Processor

Dr Brian Cowan & Dr Colin Dean

The ARM processor at the heart of the Archimedes can carry out integer calculations extremely quickly but, in common with other processors, it is significantly slower when dealing with real (floating point) numbers. This is because an integer calculation such as multiplying 3 by 4 can be achieved using a single ARM machine code instruction whereas calculating 3.75×4.23 needs many instructions.

To speed things up, a second processor, known as a 'floating point co-processor' can be connected to the ARM. This is a dedicated processor designed to perform such operations as floating point multiplication and the evaluation of sines, cosines and logarithms etc. This may be to single, double or extended precision. For the Archimedes there is the promise of a floating point co-processor module for the 400 series, based probably on the AT&T WE32206 chip.

With a co-processor attached, it is simply as if the instruction set of the main processor had been extended. Whenever the main processor encounters an instruction that it does not recognise, it sends it over to the co-processor for possible execution.

But what happens if there is no co-processor present? How does the machine then perform its floating point calculations? Acorn have adopted a most elegant solution. All language compilers assume that the floating point co-processor is present. In other words they use the extended instruction set. But in the absence of the co-processor a software emulator is used. This is installed as a relocatable module. The unrecognised instructions are sent to this module which then interprets them in terms of the main processor's instruction set. It makes

very little difference to the operating system if the co-processor is present or not.

To the software writer it is irrelevant whether the co-processor is there or not, the same op-codes may be used. Software can be developed without the co-processor in the knowledge that it will still run (much faster) with the co-processor present. Furthermore, future versions of the main processor may include the floating point instructions, once again, with no change in system software.

Floating point on BASIC V?

The Archimedes BASIC V interpreter however, does not make use of this scheme; it has its own floating point code. This is because in BBC BASIC floating point operations are performed to a non-standard precision. Calculations are executed to 40-bit precision and stored as 32 bits. On the other hand the floating point co-processor/emulator follows the IEEE standard, supporting single precision numbers in 24 bits, double precision in 53 bits and extended precision of 65 bits. Also, the BASIC assembler does not support the floating point mnemonics. So whereas it is possible to speed up BASIC programs with patches of assembler, it is not possible to call the floating point operations, this can only be done at the machine code level.

Using FPE in Assembler

Whereas the ARM CPU has 15 registers (R0 – R15) which can each hold a 32-bit integer value, the FPE has 8 registers (F0 – F7) for holding floating point values. In addition to the assembler instructions for operating on R0 – R15 and moving their contents to/from memory, there are a large number of floating point instructions. These are detailed in the Programmers' Reference Manual (part 2) and in

Peter Cockerell's book 'ARM Assembly Language Programming'. Unfortunately, the BASIC V assembler doesn't recognise the extra mnemonics, although it is possible to poke the op-codes directly.

Example:

Suppose we wanted the square root of 10.0. This can be found by loading one of the FPE registers with 10.0, e.g.

```
MVF F0, #10.0 \LET F0 = 10.0
```

the FPE can calculate square roots in one go, e.g. putting the result in register F1:

```
SQT F1, F0 \LET F1 = SQR(F0)
```

To transfer the result to memory, we can make one of the ARM registers, e.g. R0, point to the memory location and then use a floating point store instruction, e.g.

```
STF F1, [R0] \store F1 contents at
               address given by R0
```

This is all put into a small BASIC program:

```
10 REM > fptest
20 REM Using F-P Instructions
30 REM to calculate SQR(10)
40 REM Colin C Dean 08 02 88
50 :
60 DIM code 63
70 link = 14
80 FOR opt = 0 TO 2 STEP 2
90 P% = code
100 [OPT opt
110 EQUd &EE00810F \MVFS F0, #10.0
120 EQUd &EE409100 \SQTS F1, F0
130 ADR R0, result
140 EQUd &EC801100 \STFS F1, [R0]
150 MOV pc, link
160 .result
170 EQUd 0
180 ]
190 NEXT
200 :
210 CALL code
220 PRINT "Result: &"; ~!result
```

When this is RUN (FPE must be loaded first or 'Undefined instruction' error will occur), the result printed is &404A62C2. This is a 'single precision' real number, stored in 4 bytes:

```
bits 0 - 22    mantissa
%100 1010 0110 0010 1100 0010
```

```
bits 23 - 30    exponent
               %1000 0000
```

```
bit 31 sign    %0
```

To convert this to an intelligible form, we have to put a '1' in front of the mantissa, and divide by 2^{23} (giving 1.58113886), then multiply by 2^{127} (exponent - 127) (which is 2^1 here) and take note of the sign (0 => positive, 1 => negative). So the answer is 3.16227772 (phew).

The question is, of course, was this worth the effort? The answer in most cases is "no". BASIC can do SQR(10.0) pretty quickly on its own and since it uses 5 bytes to store real numbers it's actually more accurate. For this reason, BASIC doesn't use the FPE, and would ignore a co-processor even if fitted.

However, FPE instructions can, as we have said, be performed with much greater accuracy than BASIC if required (the 'S' after the mnemonics in the above program indicates 'single' precision - 'double' or 'extended' can be chosen instead). The compiled languages, Pascal, Fortran and 'C' do use the FPE and programs written in these languages will automatically become much faster than BASIC if a hardware co-processor is fitted.

In summary, BASIC neither uses nor supports the floating point co-processor/emulator. For compiled languages such as C, Fortran and Pascal, either the floating point co-processor or the emulator must be present. For assembly language programmers an assembler that supports the co-processor mnemonics is necessary but as yet unavailable. **A**

Using the SystemDevs Module

Dr Colin Dean

Introduction

There are 7 system devices which behave as filing systems:

Filing System N ^o	Name
13	NULL
14	PRINTER
15	SERIAL
17	VDU
18	RAWVDU
19	KBD
20	RAWKBD

By selecting these, various hardware devices can be accessed using filing system commands. The two most obvious cases where this can be useful are:

- 1) Diverting input or output when a program prompts for a filename e.g. if a program sends text to a named file, it can be diverted to the VDU instead
- 2) Writing programs to access hardware, which would otherwise need lots of *fx2 or *fx3 statements, e.g. an RS423 terminal

Generally speaking, filenames on the Archimedes are of the form

<filing system name>: <filename>

e.g. ADFS: \$.TEXT

ADFS::1.DRIVERS.EPSON

NULL: JUNK

The filing system name and colon can be omitted for files on the current filing system – '*Configure FileSystem Null' followed by a reset would allow 'JUNK' to be used to mean 'NULL: JUNK'. In practice, it is simplest to stick with ADFS as the current filing system. As far as the system devices are concerned, actual

filenames are irrelevant: 'NULL: fish' is the same as 'NULL: JUNK' or even 'NULL:'.

Each filing system recognises the *Close command, e.g. to close all RAWVDU files without affecting other filing systems, use:

*Rawvdu: Close

Avoid using PRINT# and INPUT# with system devices as they will generate nonsense: lines of text are best read via GET\$#, and written using BPUT#. Note that BPUT# defaults to adding line-feeds to strings, e.g.

BPUT#channel, "ABC"

writes "ABC" plus CHR\$10 to the file. A semicolon can be used to stop this, with an explicit carriage return if required, e.g.

```
BPUT#channel, "ABC"; \ "ABC" only
BPUT#channel, "ABC"+CHR$13; \ "ABC"
                        + CHR$13
BPUT#channel, "ABC"+CHR$13 \ "ABC"
                        + CHR$13 + CHR$10
```

NULL:

Any output sent here is thrown away, e.g.

```
SAVE "NULL: "           don't save at all
C%=OPENOUT"null:" }
BPUT#C%,"Hello"         } pretty pointless
CLOSE#C%                }
```

If a null file is opened for input you'll find it to be empty, e.g.

```
C%=OPENIN"null: "
IF EOF#C% THEN
  CLOSE#C%
  PRINT "A complete waste of time"
ENDIF
```

You can have more than 1 null file open at once.

PRINTER:

Bytes can be sent to the printer buffer by opening this device for output, or by whole-file operations (*copy, *save, etc), e.g.

```
chan%=OPENOUT"printer:"
BPUT#chan%,"This to printer"+CHR$13;
CLOSE#chan%
```

*Copy MyText Printer:

Beware: It's no good *COPYing a file with line feeds to a printer that needs carriage returns!

Only 1 printer file may be open at any one time.

I see from poking around in the module that there appear to be several printer 'types': null, sink, RS423, serial, parallel, centronics and user. I haven't figured out how they're used though!

SERIAL:

This provides access to RS423 input and output buffers. Only 1 file may be open at any one time, so if both input and output are needed you should use OPENUP rather than OPENIN and OPENOUT. There is no need to mess around with *fx2 and *fx3 settings.

Example to read and display lines of text from RS423 input:

```
ON ERROR OSCLI"serial:close": REPORT:
PRINT: END
C%=OPENIN"serial:"
REPEAT
PRINT GET$#C%
UNTIL FALSE
```

A nasty bug:

There is a bug (oops!) in Arthur 1.20's SERIAL device: if you open a SERIAL file and later press <reset> or <ctrl-break> without closing the file first then the machine sulks and refuses to open another SERIAL file unless you switch off and back on.

VDU and RAWVDU

Both of these send output to the VDU (even if otherwise disabled by *fx3 setting). VDU is rather like *Type in that it outputs control characters using 'I' and generates a newline for either line feed or carriage return. RAWVDU is like *Print – it outputs characters literally and so should be used instead of VDU for graphics output, etc.

Examples:

*Copy TextFile Vdu:

*Save Rawvdu: 9000 A000

Several VDU and RAWVDU files may be open at once.

KBD and RAWKBD

These can both be used to read characters from the keyboard buffer. RAWKBD can be used to read one character at a time (c.f. BASIC's GET\$ and GET), without forcing it to be displayed. KBD uses OS_Word so that input is line buffered (<ctrl-U> and <delete> have their usual effect) and each character is echoed to the screen, plus it recognises <ctrl-D> typed before <return> to mean end-of-file.

Examples:

```
file%=OPENIN"RAWKBD:"
char%=BGET#file%
CLOSE#file%

file%=OPENIN"VDU:"
index%=1
REPEAT
line$(index%)=GET$#file%
index%+=1
UNTIL EOF#file%
CLOSE#file%
```

At any one time you may have several

ADFS Disc Editor

Adrian Look

In this article Adrian describes the ADFS _DiscOp routine and puts it to good effect by writing an on-screen editor which will allow you to work on both 800 and 640 k discs.

The 'ADFS_DiscOp' Routine

This operating system call is provided by the ADFS module. It gives the user the chance to directly control the floppy disc drive's operations. It goes right down to the basics. The operations available are:

Command	Value	Registers used
Verify	0	R2, R4
Read sectors	1	R2, R3, R4
Write sectors	2	R2, R3, R4
Read track	3	R2, R3
Write track	4	R2, R3
Seek	5	R2
Step in	7	
Step out	8	

Input requirements

Each of the commands requires certain inputs. These are held in registers R1 to R4. The individual requirements of each command are listed above. The following information should be stored in the registers:

Register	Input information
R1	disc operation
R2	disc address
R3	pointer to memory
R4	length in bytes

The 'disc operation' is the value of the command to be operated as shown in the first table.

The 'disc address' indicates the head/sector/

track numbers which should be accessed by the command selected. It is given by the following formula:

$$\text{address} = ((\text{track} * \text{heads} + \text{head}) * \text{sectors_per_track} + \text{sector}) * \text{bytes_per_sector}$$

The values: heads, sectors_per_track, and bytes_per_sector are set by the ADFS and are as follows:

Format	Sectors /track	Bytes /sector	
L (640k)	16	256	1 Both use 2
D (800k)	5	1024	1 heads / 80 tracks

So, for example, if you wanted to access head 1 / sector 4 / track 58 of a 800 kbyte format disc then the calculation of the disc address would look like this:

```
heads=2          }  
sectors_per_track=5    } for 800k format  
bytes_per_sector=1024 }
```

therefore:

$$\begin{aligned} \text{address} &= ((58 * \text{heads} + 1) * \text{sectors_per_track} \\ &\quad + 4) * \text{bytes_per_sector} \\ &= \&93400 (= 603136 \text{ decimal}) \end{aligned}$$

The 'pointer to memory' value indicates, when applicable, the memory to be affected by the command selected. For example: you may wish to load the contents of a sector into memory. This could be stored at memory address &B000 – thus the pointer to memory would be set to this value (&B000)! However, it would be more advisable to reserve memory by using the DIM command in BASIC.

The length in bytes merely indicates how much information is to be processed. From the

example above, assuming 800 kbyte format, the length of a sector would be 1024 bytes. Thus to load in a sector from 1/4/58 (head/sector/track) at memory location &B000 the entry requirements would look like this:

R1 : 1 – command value (load sector)
R2 : &93400 – disc address (as above)
R3 : &B000 – pointer to memory block
R4 : 1024 – number of bytes to be loaded

Commands available

Now for an explanation on the commands which may be used:

Verify (0) reads information from the disc and checks whether it has been properly loaded.

Read Sectors (1) or Read Track (3) will read information from the disc and store it in memory.

Write Sectors (2) or Write Track (4) will put information held in memory onto the disc.

SEEK (5) will move the disc's heads to the position specified by the disc address.

STEP IN (7) or STEP OUT (8) will move the disc's heads towards 0/0/0 (head/sector/track) or away, respectively.

Exit state

When the ADFS_DiscOp is exited it will leave the registers in the following state:

Register: Exit state:

R0 0 if successful, else error pointer
R1 preserved
R2 (disc) address of next byte to be transferred
R3 (memory) pointer to next byte to be transferred
R4 number of bytes not transferred
V 1 if there was an error

(overflow flag)

Disc Utility Program

I have written a disc utility program to allow you to edit either 800 kbyte or 640 kbyte discs.

The Instructions

In the explanation of how to use the utility, the expression "click on" means move the mouse pointer onto the particular area of the screen and press <select>.

To move around the disc

(1) To move forwards or backwards by one track, head or sector at a time, set the direction by clicking on FORWARD or BACKWARD and then click on TRACK, HEAD, or SECTOR, as appropriate.

(2) If you click on the box displaying the current values of track, head and sector, you can then type in the appropriate new values followed by <return>. You have to make an entry for each of the three parameters but if you do not want to change the current value just press <return>.

If you have made any changes to the data in the current sector, the utility will first ask you whether you want to save it. Just click on either the [Y] or [N] as appropriate.

To edit the contents

Click on the particular hex or text data that you want to change and type in (in hex or text, as appropriate) the new data. To finish, either press <escape> or <adjust>.

To move around the data

Since the 800 kbyte format has too much information in each sector to display it all on the screen at one time, you have to scroll up and down through the information that is contained in the sector. This is done by either: (1) holding the <menu> button down, while sliding the pointer over the hex or text data. This scrolls the

information in the desired direction (either up or down) or (2) clicking on either <select> or <adjust> while the pointer is over the address data. This moves down or up, respectively, by sixteen rows.

Inserting a new disc

When you wish to edit a different disc, you should click on the title box and the utility will then read the size of the disc and reset itself. If this is not done, confusion may arise if the sizes do not match.

Using operating system calls

To use operating system calls within the utility, click on the box containing a star (*). Then type in all the operating system commands you want. To finish, press <return> on a blank line.

Leaving the utility

When you wish to leave the utility, click on the 'leave' box. If any change to the present sector has been made, you will be asked whether you wish to save it. Otherwise, the utility will ask you confirm that you want to leave the utility by clicking on either [Y] or [N], depending on your response.

Note: You should be very careful when editing discs – it is very easy to cause a broken directory or corrupt files – don't say we didn't warn you!

```

10 REM >$.DiskOp
20 REM *****
30 REM *      Disc Editor      *
40 REM * Adrian Philip Look *
50 REM * 23rd February 1988 *
60 REM *****
70
80 MODE 12:OFF
90 *FX 4,2
100 PROCinitialise
110 PROCset_up_screen

```

```

120 ON ERROR PROCerror
130 finished=FALSE
140 REPEAT
150   PROCinput_commands
160 UNTIL finished
170 END
180
190 DEFPROCerror
200 IF ERR<>17 THEN PRINT
   REPORT$;" at line ";ERL:END
210 *POINTER 1
220 *FX 4,2
230 VDU 28,1,30,78,7
240 PRINTTAB(0,0);
250 PROCdisplay(start,lines)
260 VDU 26
270 ENDPROC
280
290 DEFPROCinitialise
300 DIM memory% 1024
310 direction=TRUE
320 head=0:sector=0:track=0
330 tracks=80:heads=2
340 start=0:lines=23
350 changed=FALSE
360 ENDPROC
370
380 DEFPROCset_up_screen
390 VDU 19,0,24,0,0,255|
400 COLOUR 132:CLS
410 COLOUR 135:COLOUR 4
420 PROCmenu
430 PROCsize
440 PROCnext
450 *POINTER 1
460 ENDPROC
470
480 DEFPROCmenu
490 COLOUR 1
500 PROCwindow(18,1,46,1,4,
   " Archive Disc Utility - by
   Adrian Philip Look")
510 COLOUR 0
520 PROCwindow(1,3,7,1,4,
   "Forward")

```

```

530 PROCwindow(10,3,8,1,4,
               "Backward")
540 PROCwindow(21,3,5,1,4,
               "Track")
550 PROCwindow(28,3,4,1,4,
               "Head")
560 PROCwindow(34,3,6,1,4,
               "Sector")
570 PROCwindow(43,3,25,1,4,
               "Track:  Head:  Sector:  ")
580 PROCwindow(71,3,1,1,4,"")
590 PROCwindow(74,3,5,1,4,
               "Leave")
600 PROCwindow(1,5,78,26,4,"")
610 COLOUR 1
620 PRINTTAB(4,5);"Addr:"
630 PRINTTAB(24,5);
               "Hexadecimal:"
640 PRINTTAB(66,5);"ASCII:"
650 PROCupdate
660 ENDPROC
670
680 DEFPROCupdate
690 COLOUR 0
700 PRINTTAB(49,3);track;" "
710 PRINTTAB(57,3);head
720 PRINTTAB(66,3);" ";TAB
               (66,3);sector
730 COLOUR 128:COLOUR 7
740 IF direction THEN
750   PRINTTAB(1,3);"Forward"
760   ELSE
770   PRINTTAB(10,3);"Backward"
780 ENDIF
790 COLOUR 135:COLOUR 0
800 IF NOT direction THEN
810   PRINTTAB(1,3);"Forward"
820   ELSE
830   PRINTTAB(10,3);"Backward"
840 ENDIF
850 IF direction THEN dir=1
               ELSE dir=-1
860 ENDPROC
870
880 DEFPROCwindow(x,y,sx,sy,
               d,string$)
890 x=x*16:y=1024-(y+sy)*32
900 sx=sx*16:sy=sy*32
910 GCOL 0,7
920 RECTANGLE FILL x-3*d,
               y-3*d,sx+6*d,sy+6*d
930 GCOL 0,0
940 RECTANGLE FILL x-2*d,
               y-2*d,sx+4*d,sy+4*d
950 GCOL 0,7
960 RECTANGLE FILL x-d,
               y-d,sx+2*d,sy+2*d
970 PRINTTAB(x/16,(1024-y)/32
               -1);string$;
980 ENDPROC
990
1000 DEFPROCinput_commands
1010 REPEAT
1020   MOUSE x,y,buttons
1030   UNTIL buttons<>0 AND POINT
               (x,y)<>4
1040   dx=x/16:dy=32-y/32
1050   IF dy<2 AND buttons=4 THEN
1060     PROCinitialise
1070     PROCset_up_screen
1080   ENDIF
1090   IF dy>7 AND dy<30 THEN
1100     IF dx>4 AND dx<7 AND
               buttons<>2 THEN PROCpage
1110     IF dx>14 AND dx<76 THEN
1120       IF buttons=2 PROCmove
1130       IF buttons=4 PROCchange
1140     ENDIF
1150   ENDIF
1160   IF INT(dy)=3 AND dx<79 AND
               buttons=4 THEN PROCselect
1170 ENDPROC
1180
1190 DEFPROCselect
1200 IF dx>73 THEN PROCleave
1210 IF dx>70 THEN PROCstar
1220 IF dx>42 THEN PROCinput
1230 IF dx>33 THEN PROCsave
               :sector+=dir:PROCnext

```

```

1240 IF dx>27 THEN PROCsave      1580 UNTIL dy=30 OR buttons=1
      :head+=dir:PROCnext        OR (dy=23 AND size=&A00)
1250 IF dx>20 THEN PROCsave      1590 IF buttons=1 THEN PRINT
      :track+=dir:PROCnext        TAB(1,dy);:PROCdisplay
1260 IF dx>9 THEN direction      (start+dy*&10-&70,1)
      =FALSE:dx=0
1270 IF dx>1 THEN direction=TRUE 1600 *POINTER 1
1280 PROCupdate                  1610 ENDPROC
1290 ENDPROC                     1620
1300                              1630 DEFPROCchex
1310 DEFPROCchange               1640 *POINTER 0
1320 dx=INT(dx):dy=INT(dy)       1650 *FX21
1330 IF dx>59 THEN               1660 REPEAT
1340   PROCtext                  1670   pos=(dx-14)/2+(dy-7)*16
1350   ELSE                       +start
1360   IF dx<46 THEN PROCchex     1680   data=memory%?pos
1370 ENDFIF                      1690   IF pos=INT(pos) THEN
1380 ENDPROC                     data=data DIV &10 ELSE
1390                              data=data AND &F
1400 DEFPROCtext                 1700   data$=STR$(data)
1410 *POINTER 0                  1710   COLOUR 132:COLOUR 7
1420 *FX21                       1720   PRINTTAB(dx,dy);data$;
1430 REPEAT                      1730   REPEAT
1440   pos=(dx-60)+(dy-7)*16+    1740   b=INKEY(0):MOUSE x,y,
      start                      buttons
1450   data=memory%?pos:IF data   1750   UNTIL (b>64 AND b<71) OR
      <32 OR data=127            (b>47 AND b<58)
      THEN data=46              OR buttons=1
1460   data$=CHR$(data)          1760   IF buttons<>1 THEN
1470   COLOUR 132:COLOUR 7        1770   IF b<>data THEN
1480   PRINTTAB(dx,dy);data$;      changed=TRUE
1490   REPEAT                    1780   IF pos=INT(pos) THEN
1500   b=INKEY(0):MOUSE x,y,      data=(memory%?pos
      buttons                     AND &F)+EVAL("&"+
1510   UNTIL b<>-1 OR buttons=1    CHR$(b))*&10
1520   IF b<>-1 THEN              ELSE
1530   IF b<>data THEN            data=(memory%?pos
      changed=TRUE                AND &F0)+EVAL("&"+
1540   memory%?pos=b              CHR$(b))
1550   PRINTTAB(1,dy);           1820   ENDFIF
      :PROCdisplay(start         memory%?pos=data
      +dy*&10-&70,1)            1830   PRINTTAB(1,dy);:
1560   dx+=1:IF dx>75 THEN        PROCdisplay(start+
      dx=60:dy+=1              dy*&10-&70,1)
1570   ENDFIF                    1850   dx+=1:IF dx>45 THEN
      dx=14:dy+=1              1860   ENDFIF

```

```

1870 UNTIL dy=30 OR buttons=1
      OR (dy=23 AND size=&A00)
1880 IF buttons=1 THEN PRINT
      TAB(1,dy);:PROCdisplay
      (start+dy*&10-&70,1)
1890 *POINTER 1
1900 ENDPROC
1910
1920 DEFPROCmove
1930 IF size=&A00 THEN ENDPROC
1940 VDU 28,1,29,78,7
1950 dy=32-y DIV 32:dy1=dy
1960 WHILE dy>7 AND dy<30 AND
      dx>14 AND dx<76 AND
      buttons=2
1970   d--(dy<dy1)
1980   IF dy<>dy1 AND ((d=0 AND
      start>0) OR (d=1 AND
      start<&290)) THEN
1990     PROCscroll(d,1)
2000     dy1+=SGN(dy-dy1)
2010   ENDIF
2020   MOUSE x,y,buttons
2030   dx=x DIV16:dy=32-y DIV32
2040 ENDWHILE
2050 VDU 26
2060 ENDPROC
2070
2080 DEFPROCpage
2090 IF size=&A00 THEN ENDPROC
2100 IF buttons<>1 AND buttons
      <>4 THEN ENDPROC
2110 d=SGN(buttons-1)
2120 VDU 28,1,29,78,7
2130 len=ABS(d*&290-start) DIV16
2140 IF len>16 THEN len=16
2150 IF len<>0 THEN PROCscroll
      (d,len)
2160 VDU26
2170 REPEAT
2180   MOUSE x,y,buttons
2190 UNTIL buttons=0
2200 ENDPROC
2210
2220 DEFPROCscroll(d,num)
2230 where=d*(23-num)
2240 PRINTTAB(0,where);
2250 temp=0
2260 REPEAT
2270   VDU 23,7,0,d+2,0|
2280   temp+=1
2290 UNTIL temp=num
2300 IF d=0 THEN start-=&10*num
2310 PROCdisplay(start+d*&170
      ,num)
2320 IF d=1 THEN start+=&10*num
2330 ENDPROC
2340
2350 DEFPROCleave
2360 PROCsave
2370 COLOUR 4
2380 PRINT TAB(23,6);"Do you
      wish to leave? [Y] or [N]"
2390 selected=FALSE
2400 REPEAT
2410   MOUSE x,y,buttons
2420   dx=x DIV16:dy=31-y DIV32
2430   IFbuttons=4 AND dy=6 THEN
2440     IF dx=46 OR dx=53 THEN
      selected=TRUE
2450   ENDIF
2460 UNTIL selected
2470 PRINTTAB(23,6);SPC(40);
2480 IF dx=46 THEN MODE0:
      OSCLI("FX4,0"):END
2490 dx=0
2500 ENDPROC
2510
2520 DEFPROCstar
2530 *FX 15
2540 *FX 4,0
2550 VDU 28,2,29,77,7,12
2560 COLOUR 135:COLOUR 0
2570 CLS:ON
2580 LOCAL ERROR
2590 ON ERROR LOCAL PRINT'
      REPORT$

```

```

2600 REPEAT
2610   INPUT""*oscli$
2620   OSCCLI(oscli$)
2630 UNTIL oscli$=""
2640 CLS:OFF
2650 VDU 28,1,30,78,7,12
2660 PROCdisplay(start,lines)
2670 VDU 26
2680 *FX 4,2
2690 dx=0
2700 ENDPROC
2710
2720 DEFPROCinput
2730 COLOUR 4
2740 PRINTTAB(49,3);" ";TAB
      (49,3);:PROCget(2,80)
2750 IF a$<>"" THEN track=
      VAL(a$)
2760 PROCupdate:COLOUR 4
2770 PRINTTAB(57,3);" ";TAB
      (57,3);:PROCget(1,heads)
2780 IF a$<>"" THEN head=VAL(a$)
2790 PROCupdate:COLOUR 4
2800 PRINTTAB(66,3);" ";TAB
      (66,3);:PROCget(2,
      sectors_per_track)
2810 IF a$<>"" THEN sector=
      VAL(a$)
2820 PROCupdate:COLOUR 4
2830 PROCsave:PROCnext
2840 ENDPROC
2850
2860 DEFPROCget(len,limit)
2870 a$=""
2880 REPEAT
2890   b$=GET$:b=ASC(b$)
2900   IF b=127 AND LEN(a$)<>0
      THEN a$=LEFT$(a$,LEN
      (a$)-1):VDU127
2910   IF b$>="0" AND b$<="9"
      AND LEN(a$)<>len THEN
      a$+=b$:PRINTb$;
2920 UNTIL b=13
2930 IF VAL(a$)>=limit THEN
      a$=""
2940 ENDPROC
2950
2960 DEFPROCsave
2970 IF NOT changed THEN ENDPROC
2980 COLOUR 4
2990 PRINTTAB(18,6);"Do you
      wish to save changes?
      [Y] or [N]"
3000 selected=FALSE
3010 REPEAT
3020   MOUSE x,y,buttons
3030   dx=x DIV16:dy=31-y DIV32
3040   IFbuttons=4 AND dy=6 THEN
3050     IF dx=48 OR dx=55 THEN
      selected=TRUE
3060   ENDIF
3070 UNTIL selected
3080 PRINTTAB(18,6);SPC(40);
3090 IF dx=48 THEN PROCdisc_op
      (2,head,sector,track)
      :changed=FALSE
3100 ENDPROC
3110
3120 DEFPROCnext
3130 *FX 14,6
3140 IF sector=sectors_per
      _track THEN sector=0
      :head+=dir
3150 IF sector<0 THEN sector=
      sectors_per_track-1
      :head+=dir
3160 IF head=heads THEN head=0
      :track+=dir
3170 IF head<0 THEN head=heads
      -1:track+=dir
3180 IF track=tracks THEN
      track=0
3190 IF track<0 THEN track=79
3200 PROCdisc_op(1,head,sector,
      track):changed=FALSE
3210 VDU 28,1,30,78,7,12
3220 PROCdisplay(start,lines)

```

Disc Editor Utility

```

3230 VDU 26
3240 *FX 13,6
3250 dx=0
3260 ENDPROC
3270
3280 DEFFNdisc_address(head,
                        sector,track)
3290 =( (track*heads+head)*
        sectors_per_track+sector)
        *bytes_per_sector
3300
3310 DEFPROChead_sector_track
        (address)
3320 address=address DIV
        bytes_per_sector
3330 sector=address MOD
        sectors_per_track
3340 address=address DIV
        sectors_per_track
3350 head=address MOD heads
3360 track=address DIV heads
3370 ENDPROC
3380
3390 DEFPROCdisc_op(command,
        head,sector,track)
3400 B%=command
3410 C%=FNdisc_address(head,
        sector,track)
3420 D%=memory%
3430 E%=bytes_per_sector
3440 SYS "ADFS_DiscOp",,B%,C%
        ,D%,E%
3450 ENDPROC
3460
3470 DEFPROCsize
3480 SYS "ADFS_DiscOp",,1,0,
        memory%,256
3490 size=!(memory%+252) AND
        &FFFF
3500 CASE size OF
3510   WHEN &C80
3520     bytes_per_sector=1024
3530     sectors_per_track=5
3540     WHEN &A00
3550     bytes_per_sector=256
3560     sectors_per_track=16
3570     OTHERWISE
3580     ERROR 1,"Bad disc size"
3590 ENDCASE
3600 ENDPROC
3610
3620 DEFPROCdisplay(start,
        lines)
3630 COLOUR 0:COLOUR 135
3640 finish=start+lines*&10-1
3650 IF size=&A00 AND finish>
        255 THEN start=0
        :finish=255
3660 FOR memory=start TO finish
        STEP &10
3670   hex$="":ascii$=""
3680   FOR offset=0 TO 15
3690     h$=STR$~(? (memory%+
        memory+offset))
3700     IF LEN(h$)=1 THEN h$=
        "0"+h$
3710     a$=CHR$(? (memory%+
        memory+offset))
3720     IF a$<" " OR a$=CHR$
        (127) THEN a$="."
3730     hex$+=h$:ascii$+=a$
3740   NEXT offset
3750   mem$=STR$(memory)
3760   WHILE LEN(mem$)<>3
3770     mem$="0"+mem$
3780   ENDWHILE
3790   PRINT"    ";mem$;
        "    ";hex$;
        "    ";ascii$;
3800   IF memory<finish-16
        THEN PRINT ELSE memory
        =finish+1
3810 NEXT memory
3820 ENDPROC A

```

Order Form

Program Disc N^o1 ☐ N^o2 ☐ N^o3 ☐ N^o4 ☐ N^o5 ☐
 N^o6 ☐ (£3 each)

Shareware Graphics Demonstration Disc £3

Wabash 3.5" Discs Bulk pack – 10 for £16

Integrex colour screen dump **£20**

PVC Archive binder to fit 12 issues £5.50

Minerva Deltabase	£26
-------------------	-----

Minerva System Delta-Plus	£64
---------------------------	-----

Minerva Minotaur £13

Clares' Archimedes Toolkit Module £37

Clares' Graphic Writer (word-processor)	£27
---	-----

Clares' Artisan Art Package £37

Clares' Alpha-Base £46

Archimedes Programmer's Reference Manual £28

Acorn Backplane AKA 01 £42

Acorn I/O Podule AKA 10 £84

Acorn ROM Podule AKA 05 £63

Computer Concepts ROM Podule £53

with battery backup £63

32k RAM chips for ROM podule **£9.50 each**

Inter-series ROMs: Sheet £42 Chart £27 Word £42

"ARM Assembly Language Programming" £12

I enclose a cheque payable to "Norwich Computer Services" for:
(Sorry no Access or Visa facilities.)

[illegible]

All prices are inclusive of VAT, where applicable, and UK carriage.

Name _____

Address _____

*HELP Archive

What are its aims? – Archive is a subscription magazine for users of the Acorn Archimedes range of computers, which aims (1) to provide information to the user (2) to provide a forum in which we can all share ideas (3) to give the benefit of bulk buying of software (4) to allow software and hardware vendors to advertise their wares.

Is it a User Group? – No, but I would like it to have a “User Group feel” – like BEEBUG when it first started – except that Norwich Computer Services has to earn a living from the magazine – hence the order form overleaf.

However, Sue and I are not in this business to make lots of money; we enjoy the work we do and it's very satisfying to be able to provide a useful service. We only ask to make enough money to live on plus a bit to give away and then we'll be happy. (I hope it doesn't sound too trite, because it's true.)

HELP! – Could you help us, please, to make Archive a success? We don't have the muscle or the financial budget of the big magazines and cannot afford to do a vast amount of advertising, so, if you think Archive is good, please take out a full subscription (if you have not already done so) and recommend the magazine to a friend or two. If you want any more information sheets and subscription forms, please let us know.

Can we answer technical enquiries by phone?

As much as we would like to continue the policy we have always had of being available to answer all your technical enquiries by phone, we felt that if we were not careful, we would be so inundated with calls that we would not have time to get the information out to you through the magazine. For that reason, we have introduced the Technical Help Service so that those who

really need the instant access to help can purchase it. (£8 per year.) I hope you will bear with us and, if you do not feel it is worth the extra £8, please send your enquiries by post.

Do we take Access or Visa? No, I'm afraid not – mainly because of the high percentage that we would have to pay for the privilege (I think it's 6% when you first start). The other reason is that we have always had a policy of sending goods out by return of post, whenever possible, regardless of whether the cheques have cleared through the bank. The way we see it is that if you trust us by sending a cheque without the Access guarantee, it is reasonable for us to trust you by sending out the goods without waiting for the cheque to clear. Perhaps we are foolish to take the risk, but we have only had two dud cheques in three years of full time trading.

Those who help us... Although there's only two of us here, we are surrounded by a lot of people who help us in various ways, some voluntarily and some professionally, without whom we would not be able to stay in business. I don't think it is appropriate to mention them all by name, but I would just like to assure them that we are really grateful to them all. However, as I have said in most of my previous publications as a committed Christian, there is One Person who never fails us and without whom we would achieve nothing worthwhile. God supplies all our needs, and we thank and praise Him!

I hope you find Archive interesting and informative and that you will feel able to contribute your own ideas, hints and tips or even full articles. I'm afraid that we can't afford to pay vast sums for articles, but at least you'd automatically become an “HLMTHS”. Honorary Life Member, Technical Help Service!

Thanks again,



Fact-File

ACE Computing	27 Victoria Road, Cambridge, CB4 3BW. (0223 – 322559)
APL Software	7 Ascendale, Deeping St James, Peterborough, PE6 8NZ.
Brainsoft	22 Baker Street, London, W1M 1DF. (01 – 486 – 0321)
CCD Computer Services	71 Marlborough Park Avenue, Sidcup, Kent, DA15 9DL. (01 – 302 – 5427)
CJE Micros	78 Brighton Road, Worthing, W Sussex, BN11 2EN. (0903 – 213361)
Clares Micro Supplies	98 Middlewich Road, Rudheath, Northwich, Cheshire, CW9 7DA. (0606 – 48511)
Computer Concepts	Gaddesden Place, Hemel Hempstead, Herts, HP2 6EX. (0442 – 63933)
Datathorn Systems Ltd	50 Spring Grove, Loughton, Essex, IG10 4QD. (01 – 508 – 4904)
Contex Computing	15 Woodlands Close, Cople, Bedford, MK44 3UE. (02303 – 347)
EMR Ltd	14 Mount Close, Wickford, Essex, SS11 8HG. (0702 – 335747)
Fairhurst Instruments	Dean Court, Woodford Road, Wilmslow, SK9 2LT. (0625 – 525 – 694)
GEM Electronics	17 Tandragee Road, Portadown, Craigavon, BT62 3BQ.
HS Software	56, Hendrefolian Avenue, Sketty, Swansea, SA2 7NB. (0792 – 204519)
IFEL	36 Upland Drive, Plymouth, Devon, PL6 6BD. (07555 – 7286)
Intelligent Interfaces	14 Julius Close, Chandlers Ford, Eastleigh, Hants, SO5 2AB. (04125 – 61514)
LTS Ltd	Haydon House, Alcester Road, Studley, Warks, B80 7AN. (0386 – 792617)
Meadow Computers	11, London Street, Whitchurch, Hants, RG28 7LH. (025689 – 2008)
Micro Studio	83 Clay Street, Soham, Cambridge, CB7 5HL. (0353 – 721736)
Minerva Systems	69 Sidwell Street, Exeter, EX4 6PH. (0392 – 37756)
Mitre Software	26 Creechurch Lane, London, EC3A 5BA. (01 – 283 – 4646)
Musbury Consultants	8 Fairhill, Helmsshore, Rossendale, Lancs, BB4 4JX. (0706 – 216701)
Northern Educational Software	16 Dawson Lane, Bierley, Bradford, BD4 6HN.
Pineapple Software	39 Brownlea Gardens, Seven Kings, Ilford, Essex, IG3 9NL. (01 – 599 – 1476)
Quarndon Electronics	Slack Lane, Derby, DE3 3ED. (0332 – 32651)
RESOURCE	Exeter Road, Doncaster, DN2 4PY. (0302 – 63800/63784)
Solidisk Technology Ltd	17 Sweyne Avenue, Southend-on-Sea, Essex, SS2 6JQ. (0702 – 354674)
Norwich Computer Services	18 Mile End Road, Norwich, NR4 7QY. (0603 – 507057)

Archive

The Subscription Magazine for *Archimedes* users

Articles include:

- File transfer on RS423
- Attaching a 5.25" drive
- C.C.'s ROM-Link packages
- Clares' Toolkit module
- Graphics demo programs
- Archie the Music Computer
- Structuring with BASIC V
- Technical notes on BASIC V
- Help with using the ADFS
- Epson Screen Dump
- Using BBC ROM images
- BBC micro as an I/O module
- Assembly language programming
- Writing relocatable modules
- Using the WIMP environment
- Reviews of software and hardware

Technical Help Service (£8 / year)

A telephone hot-line service for immediate help with your technical problems. Any member can send written enquiries, but for a fast response use the THS!

Members' Discount: 7.5% off software from Computer Concepts, Minerva Systems and Clares Micros Supplies purchased through Norwich Computer Services.

Subscription: 12 issues £12.50 (UK)
Europe £18, Middle East £22,
America / Africa £25, Elsewhere £27.
Technical Help Service £8

Archimedes is a trademark of Acorn Computers Ltd.

* Please send copies of *Archive* magazine for one year starting from
Vol. 1.1 (Oct '87) / Vol. 1.2 / Vol. 1.3 / Vol. 1.4 / Vol. 1.5 / Vol. 1.6

* Please enrol me on the Technical Help Service for one year. (£8)

I enclose a cheque for £ _____

Name: _____

Address: _____

_____ Postcode: _____



Norwich Computer Services, 18 Mile End Road, Norwich, NR4 7QY.
Subscription – £10 per year, Technical Help Service – £8 per year